Advanced 8-bit Micro-controller with 171K Flash ROM
and Dual CAN controllers

CS8959

Advanced 8-bit Micro-controller with 171K Flash ROM
and Dual CAN controllers

Ver. 0.91

Topic of Contents

Topic of Diagrams

Advanced 8-bit Micro-controller with 171K Flash ROM and Dual CAN controllers

1. GENERAL DESCRIPTION

The CS8959 micro-controller is a single-cycle 8051 CPU core embedded device. It includes a 171K-byte internal program Flash ROM, a 256-byte internal RAM, and a 4K-byte auxiliary RAM. It also includes some advanced peripherals such as 2-channels CAN ports, 9 channels of general PWM output ports as well as a Real Time Clock (RTC) module. The Boot-Code-Free ISP (In System Programming) feature allows users to update the programming codes easily via JTAG ports or a UART port. In addition, user can debug system at real time via JTAG interface.

2. FEATURES

- Single Cycle 8051 CPU core, maximum operating clock up to 20 MHz
- Single + 5V (CS8959) power supply
- Embedded 171K-byte Flash ROM with ECC
- 4352-byte RAM with ECC
  - 256-byte Internal RAM
  - 4K-byte Auxiliary RAM
- 9 channels general PWM outputs
  - 7 channel of 8-bit supporting Programmable PWM output frequency
  - 2 channel of 8-bit supporting Programmable PWM output frequency with 4-bits flexible time base period
- Maximum 75 I/O pins
- ECC generation with 1-bit correction
- Three 16-bit Counters/Timers
- One programmable buffered clock output ports to drive peripheral devices
- Built-in one Master and two Slave I^2C ports
- Built-in SPI controller
- Hardware ISP (In-System-Programming), no Boot Code required
- Two Full-duplex UARTS
- Two CAN controllers with gateway function. Specification Version 2.0 Part A and B, each one CAN controller provides
  - Bit rate up to 1Mbit/s
  - 32 Message Objects
  - Each Message Object has its own Identifier Mask
  - 256 Bytes FIFO
  - Interrupt can be masked
  - Dispatching registers
- Multi interrupt sources
- Programmable Watch Dog Timer
- Built-in Real Time Clock (RTC)
- Dual Data Pointer (DPTR)
- Flash-ROM program code protection
- JTAG interface Debug System
- Single +5V supply voltage
- Industrial operating temperature range (-40° C ~ +85° C)
- 100-pin QFP package with RoHs compliance

Advanced 8-bit Micro-controller with 171K Flash ROM and Dual CAN controllers

3. BLOCK DIAGRAM

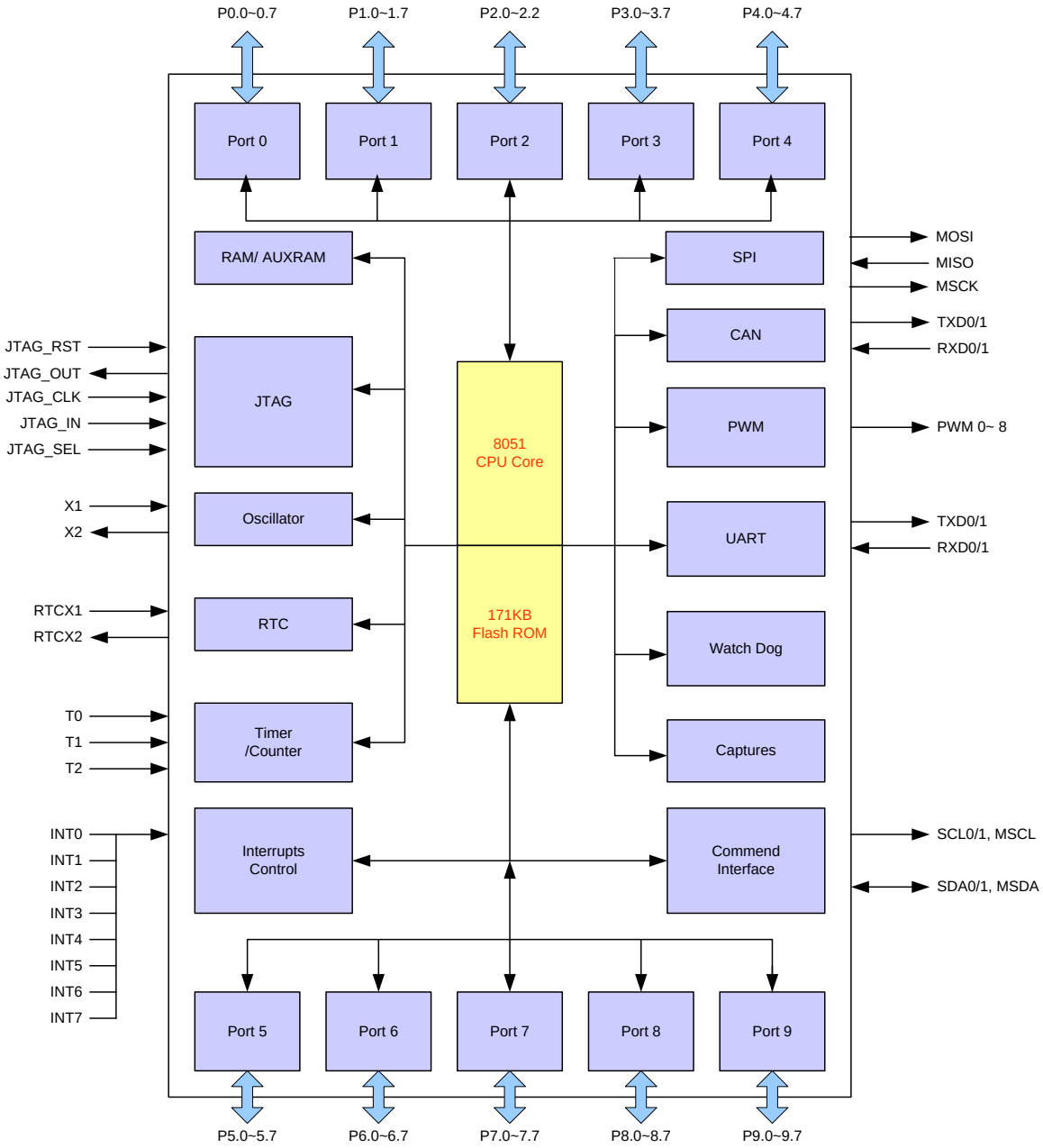


Figure-1. Block Diagram

4. PIN CONNECTION

PACKAGE TYPE

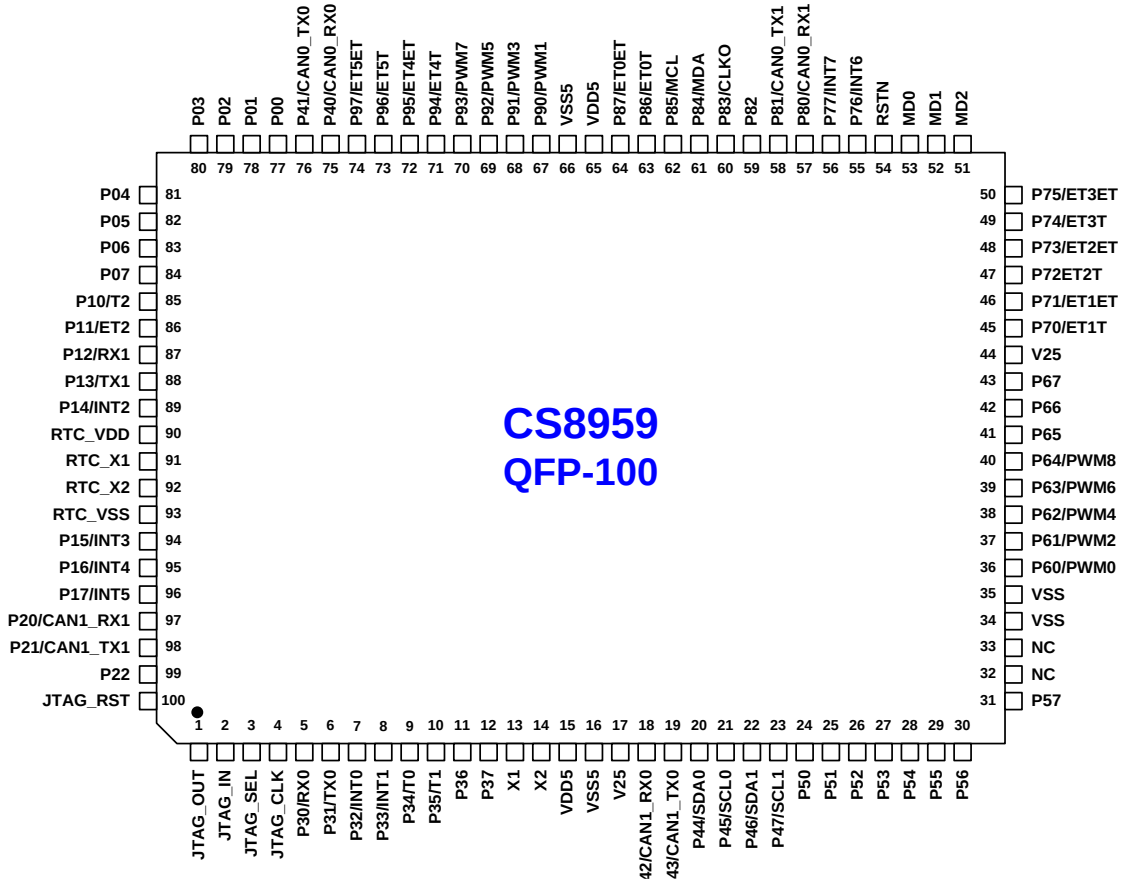


Figure-2. Package Type

5. PIN DESCRIPTION

A “CMOS pin” can be used as Input or Output mode. To use these pins as output mode, S/W needs to set the corresponding output enable control bit “Pxxoe” to 1. Otherwise, the “Pxxoe” should clear to 0. In output mode, these pins can sink and drive at least 4mA current.

An “open drain pin” means it can sink at least 4mA current but no drive current to VDD. It can be used as input or output function and need an external pull up resistor.

An “8051 standard pin” is a pseudo open drain pin. It can sink at least 4mA current when output low level, and drive at least 4mA current for 160nS when output transit from low to high, then keep drive 100uA to maintain the pin at high level. It can be used as input or output function. It need an external pull up resistor when drive heavy load device.

The pins functionality is described in the following table. All pins are one directional.
There are no three states outputs pins and internal signals.

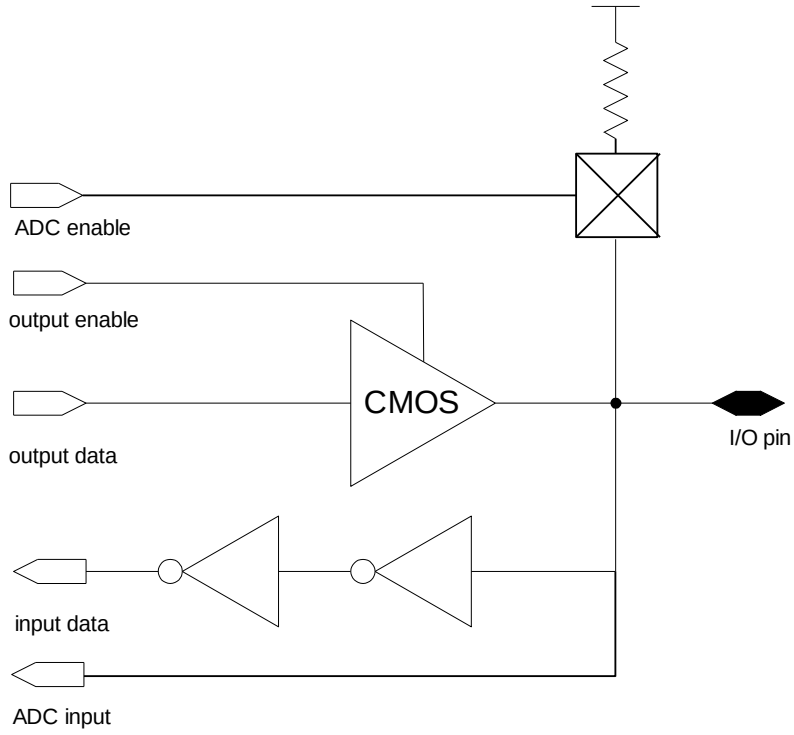
Name	Pin No.	Pin Type		Default Function	Pin Configuration	Function Description
	QFP	Type	Default			
JTAG_OUT	1	O	O	JTAG_OUT	O	JTAG output
JTAG_IN	2	I	I	JTAG_IN	U	JTAG input
JTAG_SEL	3	I	I	JTAG_SEL	U	JTAG mode select
JTAG_CLK	4	I	I	JTAG_CLK	D	JTAG clock input
P30/RX0	5	I/O	I	P30	5	General purpose I/O / UART0 serial receiver input
P31/TX0	6	I/O	I	P31	5	General purpose I/O / UART0 serial transmitter output
P32/INT0/GATE0	7	I/O	I	P32	5	General purpose I/O / External interrupt 0 / Timer0 Gate
P33/INT1/GATE1	8	I/O	I	P33	5	General purpose I/O / External interrupt 1 / Timer1 Gate
P34/T0	9	I/O	I	P34	5	General purpose I/O / Timer 0 input
P35/T1	10	I/O	I	P35	5	General purpose I/O / Timer 1 input
P36	11	I/O	I	P36	5	General purpose I/O
P37	12	I/O	I	P37	5	General purpose I/O
X1	13	I	I	X1	-	Oscillator input
X2	14	O	O	X2	-	Oscillator output
VDD5	15	-	-	VDD5	-	+5V Positive Power Supply
VSS5	16	-	-	VSS5	-	Ground
V25	17	-	-	V25	-	+2.5V Positive Power Output
P42/CAN_RX1	18	I/O	I	P42	OU	General purpose I/O / CAN1 serial transmitter output
P43/CAN_TX1	19	I/O	I	P43	OU	General purpose I/O / CAN1 serial receiver input
P44/SDA0	20	I/O	I	P44	OU	General purpose I/O / Slave IIC 0 bus serial data line
P45/SCL0	21	I/O	I	P45	OU	General purpose I/O / Slave IIC 0 bus clock line
P46/SDA1	22	I/O	I	P46	OU	General purpose I/O / Slave IIC 1 bus serial data line

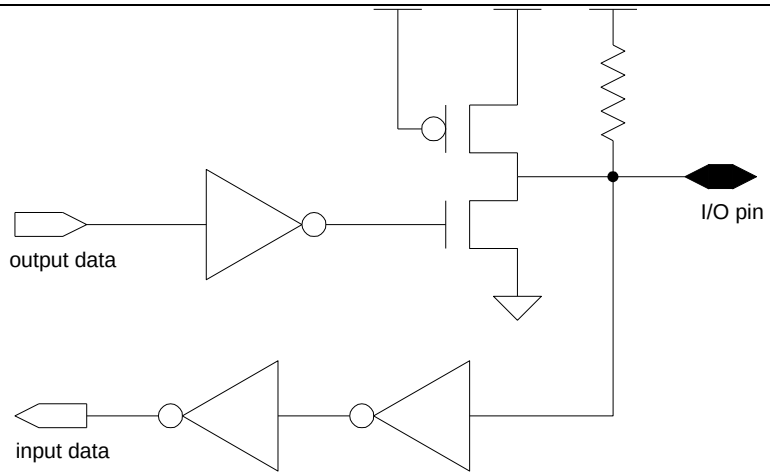
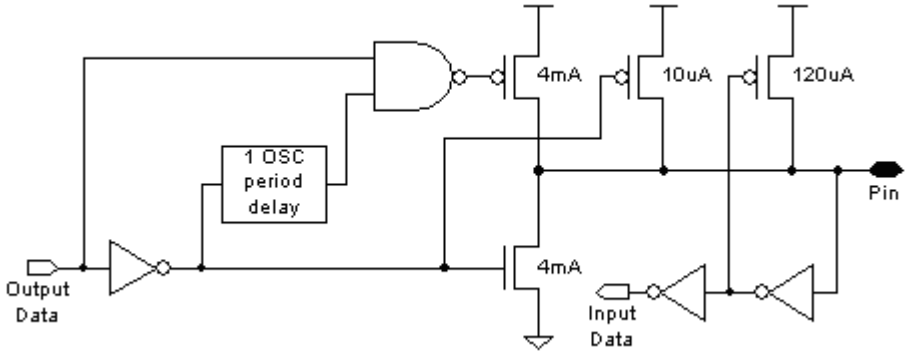
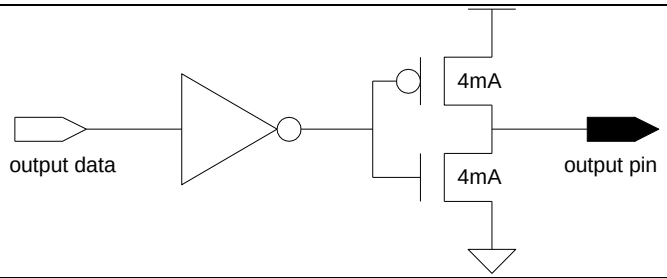
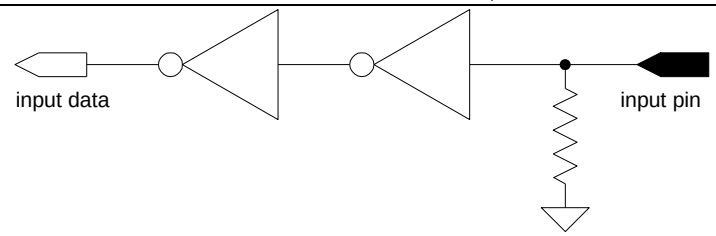
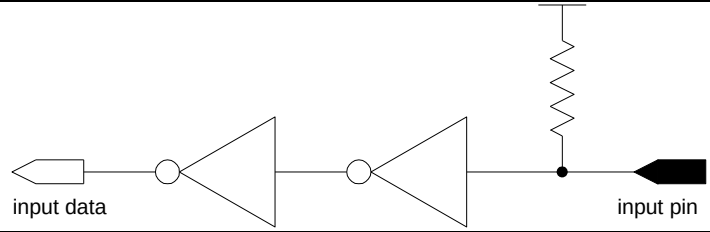
Name	Pin No.	Pin Type		Default Function	Pin Configuration	Function Description
	QFP	Type	Default			
P47/SCL1	23	I/O	I	P47	OU	General purpose I/O / Slave IIC 1 bus clock line
P50	24	I/O	I	P50	C	General purpose I/O
P51	25	I/O	I	P51	C	General purpose I/O
P52	26	I/O	I	P52	C	General purpose I/O
P53	27	I/O	I	P53	C	General purpose I/O
P54	28	I/O	I	P54	C	General purpose I/O
P55	29	I/O	I	P55	C	General purpose I/O
P56	30	I/O	I	P56	C	General purpose I/O
P57	31	I/O	I	P57	C	General purpose I/O
NC	32	-	-	NC	-	NC
NC	33	-	-	NC	-	NC
VSS	34	-	-	VSS	-	Ground
VSS	35	-	-	VSS	-	Ground
P60/PWM0	36	I/O	I	P60	OU	General purpose I/O / 8-bit PWM DAC output (with 4-BIT prescaler)
P61/PWM2	37	I/O	I	P61	OU	General purpose I/O / 8-bit PWM DAC output
P62/PWM4	38	I/O	I	P62	OU	General purpose I/O / 8-bit PWM DAC output
P63/PWM6	39	I/O	I	P63	OU	General purpose I/O / 8-bit PWM DAC output
P64/PWM8	40	I/O	I	P64	OU	General purpose I/O / 8-bit PWM DAC output
P65	41	I/O	I	P65	C	General purpose I/O
P66	42	I/O	I	P66	C	General purpose I/O
P67	43	I/O	I	P67	C	General purpose I/O
V25	44	-	-	V25	-	+2.5V Positive Power Output
P70/ET1T	45	I/O	I	P70	5	General purpose I/O / ECT1 Input.
P71/ET1ET	46	I/O	I	P71	5	General purpose I/O / ECT1 trigger Input.
P72/ET2T/SCS	47	I/O	I	P72	5	General purpose I/O / ECT2 Input / SPI chip select.
P73/ET2ET/MOSI	48	I/O	I	P73	5	General purpose I/O / ECT2 trigger Input / SPI Master output and Slave input.
P74/ET3T/MISO	49	I/O	I	P74	5	General purpose I/O / ECT3 Input / SPI Master input and Slave output.
P75/ET3ET/MSCK	50	I/O	I	P75	5	General purpose I/O / ECT3 trigger Input / SPI Master clock output.
MD2	51	I	I	MD2	D	Work mode select, shall connect to VSS25
MD1	52	I	I	MD1	D	Work mode select, shall connect to VSS25
MD0	53	I	I	MDO	D	Work mode select, shall connect to VSS25

Name	Pin No.	Pin Type		Default Function	Pin Configuration	Function Description
	QFP	Type	Default			
RSTN	54	I	I	RSTN	U	Active low reset.
P76/INT6	55	I/O	I	P76	5	General purpose I/O / External interrupt 6.
P77/INT7	56	I/O	I	P77	5	General purpose I/O / External interrupt 7
P80/CAN_RX0	57	I/O	I	P80	OU	General purpose I/O / CAN0 serial receiver input.
P81/CAN_TX0	58	I/O	I	P81	OU	General purpose I/O / CAN0 serial transmitter output.
P82	59	I/O	I	P82	5	General purpose I/O
P83/CLKO	60	I/O	I	P83	5	General purpose I/O / Oscillator Frequency Clock output
P84/MDA	61	I/O	I	P84	OU	General purpose I/O / Master IIC bus serial data line.
P85/MCL	62	I/O	I	P85	OU	General purpose I/O / Master IIC bus clock line
P86/ET0T	63	I/O	I	P86	5	General purpose I/O / ECT0 Input.
P87/ET0ET	64	I/O	I	P87	5	General purpose I/O / ECT0 trigger Input.
VDD5	65	-	-	VDD5	-	+5V Positive Power Supply
VSS5	66	-	-	VSS5	-	Ground
P90/PWM1	67	I/O	I	P90	OU	General purpose I/O / 8-bit PWM DAC output (with 4-BIT prescaler)
P91/PWM3	68	I/O	I	P91	OU	General purpose I/O / 8-bit PWM DAC output
P92/PWM5	69	I/O	I	P92	OU	General purpose I/O / 8-bit PWM DAC output
P93/PWM7	70	I/O	I	P93	OU	General purpose I/O / 8-bit PWM DAC output
P94/ET4T	71	I/O	I	P94	5	General purpose I/O / ECT4 Input.
P95/ET4ET	72	I/O	I	P95	5	General purpose I/O / ECT4 trigger Input.
P96/ET5T	73	I/O	I	P96	5	General purpose I/O / ECT5 Input.
P97/ET5ET	74	I/O	I	P97	5	General purpose I/O / ECT5 trigger Input.
P40/CAN_RX0	75	I/O	I	P40	OU	General purpose I/O / CAN0 serial receiver input.
P41/CAN_TX0	76	I/O	I	P41	OU	General purpose I/O / CAN0 serial transmitter output.
P00	77	I/O	I	P00	5	General purpose I/O
P01	78	I/O	I	P01	5	General purpose I/O
P02	79	I/O	I	P02	5	General purpose I/O
P03	80	I/O	I	P03	5	General purpose I/O
P04	81	I/O	I	P04	5	General purpose I/O
P05	82	I/O	I	P05	5	General purpose I/O
P06	83	I/O	I	P06	5	General purpose I/O
P07	84	I/O	I	P07	5	General purpose I/O
P10/T2	85	I/O	I	P10	5	General purpose I/O / External Timer 2 Input;
P11/T2EX	86	I/O	I	P11	5	General purpose I/O / External Timer 2 trigger Input

Name	Pin No.	Pin Type		Default Function	Pin Configuration	Function Description
	QFP	Type	Default			
P12/RX1	87	I/O	I	P12	5	General purpose I/O / UART1 serial receiver input
P13/TX1	88	I/O	I	P13	5	General purpose I/O / UART1 serial transmitter output
P14/INT2	89	I/O	I	P14	5	General purpose I/O / External interrupt 2
RTC_VDD	90	-	-	RTC_VDD	-	RTC Power Supply
RTC_X1	91	I	I	RTC_X1	-	RTC clock input
RTC_X2	92	O	O	RTC_X2	-	RTC clock output
RTC_VSS	93	-	-	RTC_VSS	-	RTC ground
P15/INT3	94	I/O		P15	C	General purpose I/O / External interrupt 3
P16/INT4	95	I/O	I	P16	C	General purpose I/O / External interrupt 4
P17/INT5	96	I/O	I	P17	C	General purpose I/O / External interrupt 5
P20/CAN_RX1	97	I/O	I	P20	OU	General purpose I/O / CAN1 serial receiver input.
P21/CAN_TX1	98	I/O	I	P21	OU	General purpose I/O / CAN1 serial transmitter output.
P22	99	I/O	1	P22	5	General purpose I/O
JTAG_RST	100	I	I	JTAG_RST	D	Reset JTAG module, active high reset

Note: Pin Configuration description

Pin Config.	Description	Symbol
C	Standard CMOS bi-directional pad.	

Pin Config.	Description	Symbol
OU	Open Drain with an internal pull-up resistor.	
5	Standard 8051 I/O pin	
O	Output only pad.	
D	Input only pad with an internal pull-down resistor.	
U	Input only pad with an internal pull-up resistor.	

6. MEMORY ALLOCATION

6.1 INTERNAL SPECIAL FUNCTION REGISTERS (SFR)

The SFR is a group of registers that are the same as standard 8051.

6.2 INTERNAL RAM

There are total 256 bytes internal RAM in CS8959, the same as standard 8052.

6.3 EXTERNAL SPECIAL FUNCTION REGISTERS (XFR)

The XFR is a group of registers allocated in the 8051 external RAM area 0X0F00 – 0X0FFF. These registers are used for special functions. Programs can use "MOVX" instruction to access these registers.

6.4 CAN REGISTERS

There are total 1024 bytes CAN Registers allocated in the 8051 external RAM area 0X0B00 – 0X0EFF. Programs can use "MOVX" instruction to access the CAN Registers

6.5 AUXILIARY RAM (AUXRAM)

There are total 4K auxiliary RAM allocated in the 8051 external RAM area 0X1000h – 0X1FFF. Programs can use "MOVX" instruction to access the AUXRAM.

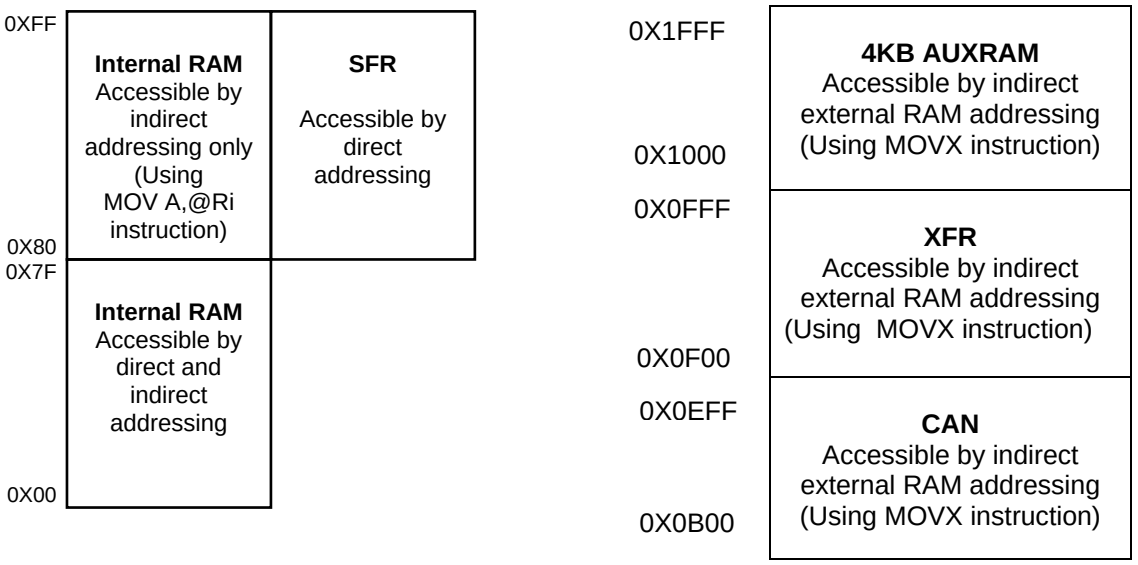


Figure-3. Auxiliary RAM

7. MEMORY MAP OF XFR (0X0F00---0X0FFF)

Reg. name	address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
IICSTUS0	0X0F01 (r)	WadrB0		SlvRWB0	SackIn0	SLVS0			
INTFLG0	0X0F03 (r)	TXBI0	RCBI0	SlvBMI0	STOPI0	ReStaI0			
INTFLG0	0X0F03 (w)			SlvBMI0	STOPI0	ReStaI0			
INTEN0	0X0F04 (w)	ETXBI0	ERCBI0	ESlvBMI0	ESTOPI0	EreStaI0			
RCBBUF0	0X0F08 (r)	Slave B 0 IIC receives buffer							
TXBBUF0	0X0F08 (w)	Slave B 0 IIC transmits buffer							
SLVBADR0	0X0F09 (w)	ENSlvB0	Slave B 0 IIC address						
ISPSLV	0X0F0B (w)	ISP Slave address							
ISPEN	0X0F0C (w)	Write 92h to enable ISP Mode							
ISPCMP1	0X0F0D (w)	ISP compared data 1 [7:0]							
ISPCMP2	0X0F0E (w)	ISP compared data 2 [7:0]							
ISPCMP3	0X0F0F (w)	ISP compared data 3 [7:0]							
DA0	0X0F20 (r/w)	Pulse width of PWM DAC 0							
DA1	0X0F21 (r/w)	Pulse width of PWM DAC 1							
DA2	0X0F22 (r/w)	Pulse width of PWM DAC 2							
DA3	0X0F23 (r/w)	Pulse width of PWM DAC 3							
DA4	0X0F24 (r/w)	Pulse width of PWM DAC 4							
DA5	0X0F25 (r/w)	Pulse width of PWM DAC 5							
DA6	0X0F26 (r/w)	Pulse width of PWM DAC 6							
DA7	0X0F27 (r/w)	Pulse width of PWM DAC 7							
DA8	0X0F28 (r/w)	Pulse width of PWM DAC 8							
RESERVED	0X0F29 (r/w)	-	-	-	-	-	-	-	-
RESERVED	0X0F2A (r/w)	-	-	-	-	-	-	-	-
PFC	0X0F2B (r/w)	PFC13	PFC12	PFC11	PFC10	PFC03	PFC02	PFC01	PFC00
PADMOD0	0X0F50 (r/w)	SPI_MOD							DA8E
PADMOD1	0X0F51 (r/w)	DA7E	DA6E	DA5E	DA4E	DA3E	DA2E	DA1E	DA0E
PADMOD2	0X0F52 (r/w)	HIIC1E	MIICEN	HIIC0E	CKOE	CAN0EN	CAN0EN1	CAN1EN	CAN1EN1
SEL	0X0F55 (w)		TRTC	WKOSC	LPS	CKSL1	CKSL2		
OPTION	0X0F56 (w)	PWMF	DIV253		ENSCL1	ENSCL0			
IICSTUS1	0X0F91 (w)	WadrB1		SlvRWB1	Sackin1	SLVS1			
INTFLG1	0X0F93 (r)	TXBI1	RCBI1	SlvBMI1	STOPI1	ReStaI1			
INTFLG1	0X0F93 (w)			SlvBMI1	STOPI1	ReStaI1			
INTEN1	0X0F94 (w)	ETXBI1	ERCBI1	ESlvBMI1	ESTOP1	Erstal1			
RCBBUF1	0X0F98 (r)	SlaveB 1 IIC receives buffer							
TXBBUF1	0X0F98 (w)	SlaveB 1 IIC transmits buffer							

Reg. name	address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SLVBADR1	0X0F99 (w)	EnslvB1	SlaveB 1 IIC address						
EXINTEN	0X0F9A (w)	INT2EN					INT5EN	INT4EN	INT3EN
EXINTFLG	0X0F9B (r/w)	INT2FLG					INT5FLG	INT4FLG	INT3FLG
ECCINTEN	0X0F9C (r/w)	PUSEN			PRGEN	D256EN	D4KEN	CAN0EN	CAN1EN

Reg. name	address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
ECCINTFLG	0X0F9D (r/w)				PRGF	D256F	D4KF	CAN0F	CAN1F
POWER	0X0F9E (r/w)	EN	FLAG				PRSTEN	CRSTEN	
PSCON	0X0F9F (r/w)	-	SWBSEL[2:0]			CAHEN			
RTCCON	0X0FE0 (w)	RTC register address						-	
RTCDAT	0X0FE1 (r/w)	RTC Data[7:0]							

7.1 Chip CONFIGURATION

The Chip Configuration registers define configuration of the chip and function of the pins.

PADMOD0 (r/w): (XFR 0X0F50) Pad mode control registers 0. (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SPI_MOD	-	-	-	-	-	-	DA8E
Default	0	-	-	-	-	-	-	0

SPI_MOD = 1 → Pin “P7.2/ET2T/SCS” is SPI_SCS; pin “P7.3/ET2ET/MOSI” is SPI_MOSI.
→ Pin “P7.4/ET3T/MISO” is SPI_MISO; pin “P7.5/ET3ET/MSCK” is SPI_MSCK.
= 0 → Pin “P7.2/ET2T/SCS” is P7.2; pin “P7.3/ET2ET/MOSI” is P7.3.
→ Pin “P7.4/ET3T/MISO” is P7.4; pin “P7.5/ET3ET/MSCK” is P7.5.

DA8E = 1 → Pin “P6.4/PWM8” is PWM8.
= 0 → Pin “P6.4/PWM8” is P6.4.

PADMOD1 (r/w) : (XFR 0X0F51) Pad mode control registers 1. (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DA7E	DA6E	DA5E	DA4E	DA3E	DA2E	DA1E	DA0E
Default	0	0	0	0	0	0	0	0

DA7E = 1 → Pin “P9.3/PWM7” is PWM7.
= 0 → Pin “P9.3/PWM7” is P9.3.
DA6E = 1 → Pin “P6.3/PWM6” is PWM6.
= 0 → Pin “P6.3/PWM6” is P6.3.
DA5E = 1 → Pin “P9.2/PWM5” is PWM5.
= 0 → Pin “P9.2/PWM5” is P9.2.
DA4E = 1 → Pin “P6.2/PWM4” is PWM4.
= 0 → Pin “P6.2/PWM4” is P6.2.
DA3E = 1 → Pin “P9.1/PWM3” is PWM3.
= 0 → Pin “P9.1/PWM3” is P9.1.
DA2E = 1 → Pin “P6.1/PWM2” is PWM2.
= 0 → Pin “P6.1/PWM2” is P6.1.
DA1E = 1 → Pin “P9.0/PWM1” is PWM1.
= 0 → Pin “P9.0/PWM1” is P9.0.
DA0E = 1 → Pin “P6.0/PWM0” is PWM0.
= 0 → Pin “P6.0/PWM0” is P6.0.

PADMOD2 (r/w) : (XFR 0X0F52) Pad mode control registers 2. (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	HIIC1E	MIICEN	HIIC0E	CKOE	CAN0EN0	CAN0EN1	CAN1EN0	CAN1EN1
Default	0	0	0	0	0	0	0	0

HIIC1E = 1 → Pin "SCL1/P4.7" is SCL1; pin "HSDA1/P4.6" is SDA1.
= 0 → Pin "SCL1/P4.7" is P4.7; pin "SDA1/P4.6" is P4.6.
MIICEN = 1 → Pin "MCL/P8.5" is MCL; pin "MDA/P8.4" is MDA.
= 0 → Pin "MCL/P8.5" is P8.5; pin "MDA/P8.4" is P8.4.
HIIC0E = 1 → Pin "SCL0/P4.5" is SCL0. Pin "SDA0/P4.4" is SDA0.
= 0 → Pin "SCL0/P4.5" is P4.5. Pin "SDA0/P4.4" is P4.4.
CKOE = 1 → Pin "P8.3/CLKO" is CLKO.
= 0 → Pin "P8.3/CLKO" is p8.3.
CAN0EN0 = 1 → Pin "CAN_RX0/P4.0" is CAN0_RX0; pin "CAN_TX0/P4.1" is CAN0_TX0.
= 0 → Pin "CAN_RX0/P4.0" is P4.0; pin "CAN_TX0/ P4.1" is P4.1.
CAN0EN1 = 1 → Pin "CAN_RX0/P8.0" is CAN1_RX0; pin "CAN_TX0/P8.1" is CAN1_TX0.
= 0 → Pin "CAN_RX0/P8.0" is P8.0; pin "CAN_TX0/ P8.1" is P8.1.
CAN1EN0 = 1 → Pin "CAN_RX1/P4.2" is CAN1_RX0; pin "CAN_TX1/P4.3" is CAN1_TX0.
= 0 → Pin "CAN_RX1/P4.2" is P4.2; pin "CAN_TX1/ P4.3" is P4.3.
CAN1EN1 = 1 → Pin "CAN_RX1/P2.0" is CAN1_RX1; pin "CAN_TX1/P2.1" is CAN1_TX1.
= 0 → Pin "CAN_RX1/P2.0" is P2.0; pin "CAN_TX1/ P2.1" is P2.1.

OPTION (r/w) : (XFR 0X0F56) chip option configuration (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	PWMF	DIV253	-	ENSCL1	ENSCL0	-	-	-
Default	0	0	-	0	0	-	-	-

PWMF = 1 → Selects system clock/512 as PWM clock frequency.
= 0 → Selects system clock/1024 as PWM clock frequency.
DIV253 = 1 → PWM pulse width is 253-step resolution.
= 0 → PWM pulse width is 256-step resolution.
ENSCL1 = 1 → Enable SlaveB 1 IIC block to hold SCL pin low while CS8959 is unable to catch-up with the external master's speed.
ENSCL0 = 1 → Enable SlaveB 0 IIC block to hold SCL pin low while CS8959 is unable to catch-up with the external master's speed.

PSCON(w/r) : (XFR 0X0F9F) Power save mode control (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	-	SWBSEL2	SWBSEL1	SWBSEL0	CAHEN	-	-	-
Default	-	0	0	0	0	-	-	-

SWBSEL[2:0] → Determine the time period CPU switch back from stop mode after switch back condition.

SWBSEL [2:0]	Delay time after switch back condition.
000	2,000 x system clock
001	20,000 x system clock
010	40,000 x system clock
011	100,000 x system clock

SWBSEL [2:0]	Delay time after switch back condition.
100	200,000 x system clock
101	400,000 x system clock
110	600,000 x system clock
111	1,000,000 x system clock

CAHEN = 1 → Enable internal 1 byte flash cache.
= 0 → Disable internal 1 byte flash cache.

8. CS8959 SFR ALLOCATION

8.1 INTRODUCTION

CS8959 is an ultra high performance, speed optimized 8-bit embedded controller dedicated for operation with fast (typically on-chip) and slow (off-chip) memories. The chip has been designed with a special concern about performance to power consumption ratio. CS8959 is 100% binary-compatible with the industry standard 8051 8-bit micro-controller. The CS8959 is Hardware configure where internal data and program buses are separated. CS8959 has pipelined RISC architecture that is 6.7 times faster in typical applications compared to standard architecture. This performance can also be exploited to great advantage in low power applications where the core can be clocked more slowly than the original implementation for no performance penalty.

8.2 INTERNAL DATA MEMORY & SFRS ALLOCATION

The picture below shows the Internal Memory and Special Function Registers (SFR) map.

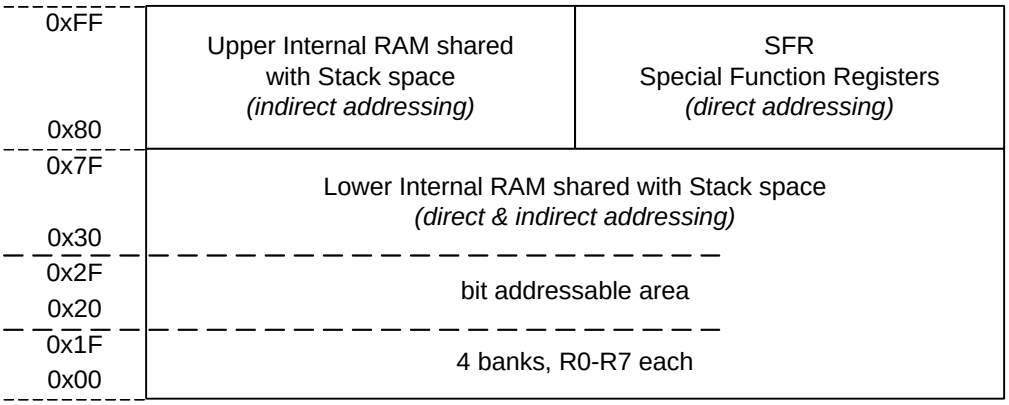


Figure-4. SFR Map

The lower internal RAM consists of four register banks with eight registers each, a bit addressable segment with 128 bits (16 bytes) begins at 0X20, and a scratch pad area with 208 bytes. With the indirect addressing mode range 0X80 to 0XFF highest 128 bytes of the internal memory is addressed. With the direct addressing mode range 0X80 to 0XFF the SFR memory area is accessed.

SFR Memory Map

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F	
0XF8	EXIP	MD0	MD1	MD2	MD3	MD4	MD5	ARCON	0XFF
0XF0	B				I2CMSA	I2CMCR	I2CMBUF	I2CMTP	0XF7
0XE8	EXIE		MXAX		SPICR	SPIMR	SPIST	SPIDATA	0XEF
0XE0	ACC	T2CON5	RLDL_5	RLDH_5	TH2_5	TL2_5			0XE7
0XD8	WDCON	T2CON4	RLDL_4	RLDH_4	TH2_4	TL2_4			0XDF
0XD0	PSW	T2CON3	RLDL_3	RLDH_3	TH2_3	TL2_3			0XD7
0XC8	T2CON	T2MOD	RCAP2L	RCAP2H	TL2	TH2			0XCF
0XC0	SCON1	SBUF1			PMR	STATUS	Reserved	TA	0XC7
0XB8	IP	T2CON2	RLDL_2	RLDH_2	TH2_2	TL2_2			0XBF
0XB0	P3	T2CON1	RLDL_1	RLDH_1	TH2_1	TL2_1			0XB7
0XA8	IE	T2CON0	RLDL_0	RLDH_0	TH2_0	TL2_0			0XAF
0XA0	P2	P4	P5	P6	P7	P8	P9		0XA7
0X98	SCON0	SBUF0	DPXR	ESP		ACON		LVDT	0X9F
0X90	P1	EXIF	WTST	DPX		DPX1			0X97
0X88	TCON	TMOD	TL0	TL1	TH0	TH1	CKCON		0X8F
0X80	P0	SP	DPL	DPH	DPL1	DPH1	DPS	PCON	0X87
	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F	

Default value:

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F	
0XF8	00H	00H	00H	00H	00H	00H	00H	00H	0XFF
0XF0	00H				00H	00H	00H	FFH	0XF7
0XE8	00H		00H		20H	00H	00H	00H	0XEF
0XE0	00H	00H	00H	00H	00H	00H			0XE7
0XD8	00H	00H	00H	00H	00H	00H			0XDF
0XD0	00H	00H	00H	00H	00H	00H			0XD7
0XC8	00H	00H	00H	00H	00H	00H			0XCF
0XC0	00H	00H			00H	00H	00H	00H	0XC7
0XB8	00H	00H	00H	00H	00H	00H			0XBF
0XB0	FFH	00H	00H	00H	00H	00H			0XB7
0XA8	00H	00H	00H	00H	00H	00H			0XAF
0XA0	FFH	FFH	FFH	FFH	FFH	FFH	FFH		0XA7
0X98	00H	00H	00H	10H		00H		20h	0X9F
0X90	FFH	00h	00H	00H		00H			0X97
0X88	00H	00H	00H	00H	00H	00H	03H		0X8F
0X80	FFH	07H	00H	00H	00H	00H	00H	00H	0X87
	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F	

9. SPECIAL SFR DEFINITION

9.1 PROGRAM WAIT STATES REGISTER

Wait States register holds the information about Program Memory access time. It allows the CS8959 core operation with fast (on-chip) and slow (off-chip) Program Memories.

9.1.1 Wait-states Cycle Register (default 0X00)

WTST(w/r) : (SFR 0X92) Wait-states Cycle Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	-	-	-	-	WTST.3	WTST.2	WTST.1	WTST.0
Default	-	-	-	-	0	0	0	0

Note: These bits are considered during program fetches and MOVC instructions only.
Minimal read cycle takes 1 clock period and maximal 16 clock periods

9.1.2 WTST Register Values

WTST value WTST[3:0]	Access time [clk]
15	16
14	15
…	…
1	2
0	1

WTST[3:0] should be not less than 1 at this version

9.2 FLAT/LARGE MODEL SWITCHING

Switching between LARGE and FLAT modes is performed by appropriate writes into ACON(0X9D) register. This register is timed access register, prevents from accidental change of working mode.

9.2.1 Address Control Register

ACON(w/r) : (SFR 0X9D) Address Control Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	-	-	-	-	-	SA	AM1	AM0
Default	-	-	-	-	-	0	0	0

SA ----- Extend Stack Address Mode indicator. This bit is read only:
0 ----- All instructions utilize the 8-bit 8051 stack pointer SP(0X81) resides in internal data memory.
1 ----- All instructions utilize the 16-bit stack pointer formed by ESP:SP register (0X9B:OX81). Stack resides in external data memory.
AM[1:0] ----- Address Mode Control bits. These bits establish the addressing mode for the micro controller. They can only be modified by the Timed access register procedure.

9.2.2 Address Control

AM[1:0]	Addressing Mode
00	16-bit Addressing Mode-LARGE
1X	24-bit contiguous Addressing Mode-FLAT

- Unimplemented bit. Read as 0 or 1.

To switch between modes the following instructions should be performed:

- *MOV TA ,#0XAA*
- *MOV TA,#0X55*
- *MOV ACON ,#0X02;Switch to FLAT mode*

Or

- *MOV TA ,#0XAA*
- *MOV TA ,#0X55*
- *MOV ACON,#0X00;Switch to LARGE Mode*

9.2.3 Stack Pointer Register

SP(w/r) : (SFR 0X81) Stack Pointer Register (default 0X03)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SP.7	SP.6	SP.5	SP.4	SP.3	SP.2	SP.1	SP.0
Default	0	0	0	0	0	0	1	1

SP(0X81) register holds the low order 8-bit address used for addressing the Stack Memory. When processor is in LARGE or FLAT mode. After reset SP has 0X07 value.

Extended Stack Pointer Register

ESP(w/r) : (SFR 0X9B) Extended Stack Pointer Register (default 0X10)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	ESP.7	ESP.6	ESP.5	ESP.4	ESP.3	ESP.2	ESP.1	ESP.0
Default	0	0	0	1	0	0	0	0

ESP(0X9B) register holds the eight high order address bits used for addressing the Stack Memory when processor is in FLAT mode. After reset ESP has 0X00 value.

9.3 DATA POINTER EXTENDED REGISTERS

Data Pointer Extended register DPX, DPX1, MXAX, and DPXR hold the most significant part of memory Address during access to data located above 64 kB in LARGE and FLAT mode. After reset DPX, DPX1, MXAX, DPXR have 0X00 value.

9.3.1 Data Pointer Extended Register

DPX(w/r) : (SFR 0X93) Data Pointer Extended Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DPX.7	DPX.6	DPX.5	DPX.4	DPX.3	DPX.2	DPX.1	DPX.0
Default	0	0	0	0	0	0	0	0

9.3.2 Data Pointer Extended 1 Register

DPX1(w/r) : (SFR 0X95) Data Pointer Extended 1 Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DPX1.7	DPX1.6	DPX1.5	DPX1.4	DPX1.3	DPX1.2	DPX1.1	DPX1.0
Default	0	0	0	0	0	0	0	0

9.3.3 MOVX @Ri Extended High Register

MXAX(w/r) : (SFR 0XEA) MOVX @Ri Extended High Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	MXAX.7	MXAX.6	MXAX.5	MXAX.4	MXAX.3	MXAX.2	MXAX.1	MXAX.0
Default	0	0	0	0	0	0	0	0

9.3.4 MOVX @Ri Extended Middle Register

DPXR(w/r) : (SFR 0X9A) MOVX @Ri Extended Middle Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DPXR.7	DPXR.6	DPXR.5	DPXR.4	DPXR.3	DPXR.2	DPXR.1	DPXR.0
Default	0	0	0	0	0	0	0	0

During MOVX instruction using DPTR/DPTR1 registers, the most significant part of address XRAMADDR[23:16] is always equal to DPX(0X93)/DPX1(0X95) contents. During MOVX instruction using R0 or R1 registers, the most significant part of address XRAMADDR[23:16] is always equal to MXAX(0XEA) contents and XRAMADDR[15:8] is always equal to DPXR(0X9A) contents, which is different from traditional 8051 MCU. These rules apply for FLAT and LARGE modes.

9.4 RESERVED REGISTER

User should not write any value to this register (0XC6).

9.5 DATA POINTER REGISTER

Dual data pointer registers are implemented to speed up data block copying. DPTR and DPTR1 are located at four SFR addresses. Active DPTR register is selected by SEL bit (0X86.0). If SEL bit is equal to 0 then DPTR(0X83:0X82) is selected otherwise DPTR1 (0X85:0X84).

9.5.1 Data Pointer Register DPTR (default 0X0000)

DPH(w/r) : (SFR 0X83)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DPH.7	DPH.6	DPH.5	DPH.4	DPH.3	DPH.2	DPH.1	DPH.0
Default	0	0	0	0	0	0	0	0

DPL(w/r) : (SFR 0X82)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DPL.7	DPL.6	DPL.5	DPL.4	DPL.3	DPL.2	DPL.1	DPL.0
Default	0	0	0	0	0	0	0	0

9.5.2 Data Pointer 1 Register DPTR1 (default 0X0000)

DPH1(w/r) : (SFR 0X85)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DPH1.7	DPH1.6	DPH1.5	DPH1.4	DPH1.3	DPH1.2	DPH1.1	DPH1.0
Default	0	0	0	0	0	0	0	0

DPL1(w/r) : (SFR 0X84)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DPL1.7	DPL1.6	DPL1.5	DPL1.4	DPL1.3	DPL1.2	DPL1.1	DPL1.0
Default	0	0	0	0	0	0	0	0

9.5.3 Data pointer Select Register

DPS(w/r) : (SFR 0X86) Data pointer Select Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	ID1	ID0	TSL	-	-	-	-	SEL
Default	0	0	0	-	-	-	-	0

ID[1:0] - Increment/decrement function select. See table below.
TSL - Toggle select enable. When set, this bit allows the following DPTR related instructions to toggle the SEL bit following execution of the instruction:

- *INC DPTR*
- *MOV DPTR, #data16/#data24*
- *MOVC A, @A+DPTR*
- *MOVX @DPTR, A*
- *MOVX A, @DPTR*
- *When TSL=0, DPTR related instructions will not affect the state of the SEL bit.*

- - Unimplemented bit. Read as 0 or 1

DPTR0, DPTR 1 operation

ID1	ID0	SEL=0	SEL=1
0	0	INC DPTR	INC DPTR 1
0	1	DEC DPTR	INC DPTR 1
1	0	INC DPTR	DEC DPTR 1
1	1	DEC DPTR	DEC DPTR 1

Selected data pointer register is used in the following instructions:

- *MOVX @DPTR,A*
- *MOVX A, @DPTR*
- *MOVC A, @A+DPTR*
- *JMP @A+DPTR*
- *INC DPTR*
- *MOV DPTR, #data16/#data24*

9.6 CLOCK CONTROL REGISTER-STRETCH BITS

Clock control register CKCON(0X8E) contains MD[2:0] bits (STRETCH bits) hold the information about XRAMRD and XRAMWR pulses width.

9.6.1 Clock Control Register

CKCON(w/r) : (SFR 0X8E) Clock Control Register (default 0X03)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	WD1	WD0	T2M	T1M	T0M	MD2	MD1	MD0
Default	0	0	0	0	0	0	1	1

XRAMRD and XRAMWR signals are activated during MOVX instruction. The purpose of STRETCH bits is adjusting of communication speed with I/O devices such as slow RAM etc. After reset STRETCH bits have 0X03 value. It means that slow devices should work properly. If needed user can change stretch value, to speed up/slow down software execution. Stretch value can be changed any time during program execution (e.g. between MOVX to different speed devices).

9.6.2 Clock Control Period

MD[2:0]	Pulse width[clk]
7	8
6	7
…	…
1	2
0	1

9.7 TIMED ACCESS REGISTERS

Timed access registers have built in mechanism preventing them from accidental Writes. To do a correct write to such registers the following sequence has to be applied:

- *MOV TA , #0XAA*
- *MOV TA , #0X55*
- *;Any direct addressing instruction writing timed access register.*

The time elapsed between first, second, and third operation does not matter (any number of Program Wait Sates is allowed). The only correct sequence is required. Any third instruction causes Protection mechanism to be turned on. This means that time protected register is opened for write only for single instruction. Reading from such register is never protected. Time access registers are listed in table below.

9.7.1 Timed Access Registers

Register name	Description	Default
ACON(0X9D)	Address control register	0X00
WDCON(0XD8)	Watchdog configuration	0X00

10. INTERRUPT SYSTEM

CS8959 has implemented two levels interrupt priority control. Each external interrupt can be in high or low level priority group by setting or clearing a bit in the IP(0XB8), and EIP(0XF8) registers. External interrupt pins are activated at low level or by a falling edge Interrupt requests are sampled each system clock at the rising edge of CLK.

10.1 INTERRUPT VECTOR

Interrupt Flag	Function	Active level/edge	Flag resets	Vector	Natural priority
IE0	PIN INT0	Low/Falling	Hardware	0X03	0
TF0	Timer 0	-	Hardware	0X0B	1
IE1	PIN INT1	Low/Falling	Hardware	0X13	2
TF1	Timer 1	-	Hardware	0X1B	3
TI0&RI0	UART0	-	Software	0X23	4
TF2	Timer2	-	Software	0X2B	5
TI1&RI1	UART1	-	Software	0X33	6
I2CMF	MI2C	-	Software	0X3B	7
AFIF	AFI	Low	Hardware	0X43	8
CAN0I	CAN0I	Low	Hardware	0X4B	9
CAN1I	CAN1I	Low	Hardware	0X53	10
WDIF	Watchdog	-	Software	0X5B	11
INT6F	PIN INT6 SPI INT	Falling	Software	0X63	12
INT7F	PIN INT7	Falling	Software	0X6B	13
CANBUSI	CANBUSI	Falling	Software	0X73	14

-----AFI: affixation function interrupt

All interrupts related registers are listed in this chapter. Each interrupt vector can be individually enabled or disabled by setting or clearing the corresponding bit in the IE(0XA8), T2CON(0XC8), or EXIE(0XE8). The IE contains global disable bit called EA.

10.1.1 Interrupt Enable Register

IE(w/r) : (SFR 0XA8) Interrupt Enable Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	EA	ES1	ET2	ES0	ET1	EX1	ET0	EX0
Default	0	0	0	0	0	0	0	0

EA - Enable global interrupts
EX0 - Enable INT0 interrupts
ET0 - Enable Timer 0 interrupts
EX1 - Enable INT1 interrupts
ET1 - Enable Timer 1 interrupts
ES0 - Enable UART0 interrupts
ET2 - Enable Timer 2 interrupts
ES1 - Enable UART1 interrupts

Each interrupt source can be individually programmed to one of the two priority levels by setting or clearing a bit in the IP(0XB8) or EXIP(0XF8) registers.

10.1.2 Interrupt Priority Register

IP(w/r) : (SFR 0XB8) Interrupt Priority Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	-	PS1	PT2	PS0	PT1	PX1	PT0	PX0
Default	-	0	0	0	0	0	0	0

PX0 - INT0 priority level control (at 1-high level)
PT0 - Timer 0 priority level control (at 1-high level)
PX1 - INT1 priority level control (at 1-high level)
PT1 - Timer 1 priority level control (at 1-high level)
PS0 - UART0 priority level control (at 1-high level)
PT2 - Timer 2 priority level control (at 1-high level)
PS1 - UART1 priority level control (at 1-high level)
- - Unimplemented bit. Read as 0 or 1.

The DI2CM, external interrupts AFI, INT6, INT7, CAN0I interrupt, CAN1I interrupt, CANBUSI interrupt, and Watchdog interrupt control bits are located in EXIE(0XE8), EXIP(0XF8), EXIF(0X91) and WDCON(0XD8) registers.

10.1.3 External Interrupt Enable Register

EXIE(w/r) : (SFR 0XE8) External Interrupt Enable Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	ECANBUS	EINT7	EINT6	EWDI	ECAN1	ECAN0	EAFI	EI2CM
Default	0	0	0	0	0	0	0	0

EI2CM - Enable Master I^2C (DI2CM) interrupt.
EAFI - Enable Affixation Functions Interrupts. Affixation Functions Interrupts include slave I^2C interrupt, ECT interrupts, external interrupt 2, external interrupt 3, external interrupt 4, external interrupt 5, RTC interrupt, power warning interrupt, and HC_ER interrupts. HC_ER interrupts include 256 bytes ram error interrupt, program flash error interrupt, 4k AUXRAM error interrupt, CAN0 ram error interrupt, and CAN1 ram error interrupt. If use one of these interrupts must enable this bit first.
ECAN0 - Enable CAN0 interrupt.
ECAN1 - Enable CAN1 interrupt.
EWDI - Enable Watchdog interrupt.
EINT6 - Enable INT6 interrupt.
EINT7 - Enable INT7 interrupt.
ECANBUS - Enable CANBUSI interrupt.

10.1.4 External Interrupt Priority Register

EXIP(w/r) : (SFR 0XF8) External Interrupt Priority Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	PCANBUS	PINT7	PINT6	PWDI	PCAN1	PCAN0	PAFI	PI2CM
Default	0	0	0	0	0	0	0	0

PI2CM - Master I^2C (DI2CM) priority level control (at 1-high level)
PAFI - AFI priority level control (at 1-high level)
PCAN0 - CAN0 priority level control (at 1-high level)
PCAN1 – CAN1 priority level control (at 1-high level)
PWDI - Watchdog priority level control (at 1-high level)
PINT6 - INT6 priority level control (at 1-high level)
PINT7 - INT7 priority level control (at 1-high level)
PCANBUS - CANBUSI priority level control (at 1-high level)

10.1.5 External Interrupt Flag Register

EXIF(w/r) : (SFR 0X91) External Interrupt Flag Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	CANBUSI	INT7F	INT6F	-	CAN1I	CAN0I	AFIF	I2CMIF
Default	0	0	0	-	0	0	0	0

I2CMIF - Master I^2C (DI2CM) interrupt flag. Must be cleared by software
AFIF - AFI interrupt flag. Cleared by hardware when processor branches to interrupt routine
CAN0I - CAN0 interrupt flag. Cleared by hardware when processor branches to interrupt routine
CAN1I - CAN1 interrupt flag. Cleared by hardware when processor branches to interrupt routine
INT6F - INT6 interrupt flag. Must be cleared by software
INT7F - INT7 interrupt flag. Must be cleared by software
CANBUSI - CANBUSI interrupt flag. Must be cleared by software
- Unimplemented bit. Read as 0 or 1.

The watchdog interrupt is located in WDCON(0XD8).The external interrupts INT0-INT1, and Timer 0, 1, 2 interrupt control bits are located in TCON(0X88), T2CON(0XC8), IE(0XA8) and IP(0XB8) registers.

10.1.6 Timer 0,1 Configuration Register

TCON(w/r) : (SFR 0X88) Timer 0,1 Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
Default	0	0	0	0	0	0	0	0

IT0 - INT0 level (at 0)/ edge (at 1) sensitivity
IT1 - INT1 level (at 0)/ edge (at 1) sensitivity
IE0 - INT0 interrupt flag. Cleared by hardware when processor branches to interrupt routine
IE1 - INT1 interrupt flag. Cleared by hardware when processor branches to interrupt routine
TF0 - Timer 0 interrupt (overflow) flag. Cleared by hardware when processor branches to interrupt routine
TF1 - Timer 1 interrupt (overflow) flag. Cleared by hardware when processor branches to interrupt routine

10.1.7 Timer 2 Configuration Register

T2CON(w/r) : (SFR 0XC8) Timer 2 Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	CT2	CPRL2
Default	0	0	0	0	0	0	0	0

TF2 - Timer 2 interrupt (overflow) flag. Must be cleared by software. The flag will not be set when either RCLK or TCLK is set.
EXF2 - Falling edge indicator on T2EX pin when EXEN2=1. Must be cleared by software.
EXEN2 - Enable T2EX pin functionality.
1 - allows capture or reload as a result of T2EX pin falling edge
0 - ignore T2EX events

10.1.8 UART0 Configuration Register

SCON0 (w/r) : (SFR 0X98) UART0 Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SM00	SM01	SM02	REN0	TB08	RB08	TI0	RI0
Default	0	0	0	0	0	0	0	0

RI0 - UART0 receiver interrupt flag. Must be cleared by software
TI0 - UART0 transmitter interrupt flag. Must be cleared by software

10.1.9 UART1Configuration Register

SCON1 (w/r) : (SFR 0XC0) UART0 Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SM10	SM11	SM12	REN1	TB18	RB18	TI1	RI1
Default	0	0	0	0	0	0	0	0

RI1 - UART1 receiver interrupt flag. Must be cleared by software
TI1 - UART1 transmitter interrupt flag. Must be cleared by software

10.1.10 External Interrupt Registers

EXINTEN(w/r) : (XFR 0X0F9A) External interrupt enable (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	EXINT2EN	-	-	-	-	EXINT5EN	EXINT4EN	EXINT3EN
Default	0	-	-	-	-	0	0	0

INT2EN = 1 → Enable external interrupt 2 from P1.4.
INT5EN = 1 → Enable external interrupt 5 from P1.7.
INT4EN = 1 → Enable external interrupt 4 from P1.6.
INT3EN = 1 → Enable external interrupt 3 from P1.5.

EXINTFLG (w/r) : (XFR 0X0F9B) External Interrupt Flag (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	INT2FLG	-	-	-	-	INT5FLG	INT4FLG	INT3FLG
Default	0	-	-	-	-	0	0	0

INT2FLG → INT2 interrupt flag. Must be cleared by software
INT5FLG → INT5 interrupt flag. Must be cleared by software
INT4FLG → INT4 interrupt flag. Must be cleared by software
INT3FLG → INT3 interrupt flag. Must be cleared by software

ECCINTEN(w/r) : (XFR 0X0F9C) Error Correction Code interrupt enable. (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	PUSEN	-	-	PRGEN	D256EN	D4KEN	CAN0EN	CAN1EN
Default	0	-	-	0	0	0	0	0

PRGEN = 1 → Enable program RAM error interrupt.
D256EN = 1 → Enable 256 bytes internal RAM error interrupt.
D4KEN = 1 → Enable 4k AUXRAM error interrupt.
CAN0EN = 1 → Enable CAN0 RAM error interrupt.
CAN1EN = 1 → Enable CAN1 RAM error interrupt.

ECCINTFLG(w/r): (XFR 0X0F9D) Error Correction Code interrupt flag. (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	-	-	-	PRGF	D256F	D4KF	CAN0F	CAN1F
Default	-	-	-	0	0	0	0	0

PRGF → The program RAM error interrupt flag. Must be cleared by software.
D256F → The 256 bytes internal RAM error interrupt flag. Must be cleared by software.
D4KEF → 4k AUXRAM ram error interrupt flag. Must be cleared by software.
CAN0F → CAN0 RAM error interrupt flag. Must be cleared by software.
CAN1F → CAN1 RAM error interrupt flag. Must be cleared by software.

In CS8959, those external interrupt, INT2-INT5, and all RAM error interrupts are shared with AFIF that interrupt vector is 0X43.

The figurate below is all interrupts diagram of CS8959.

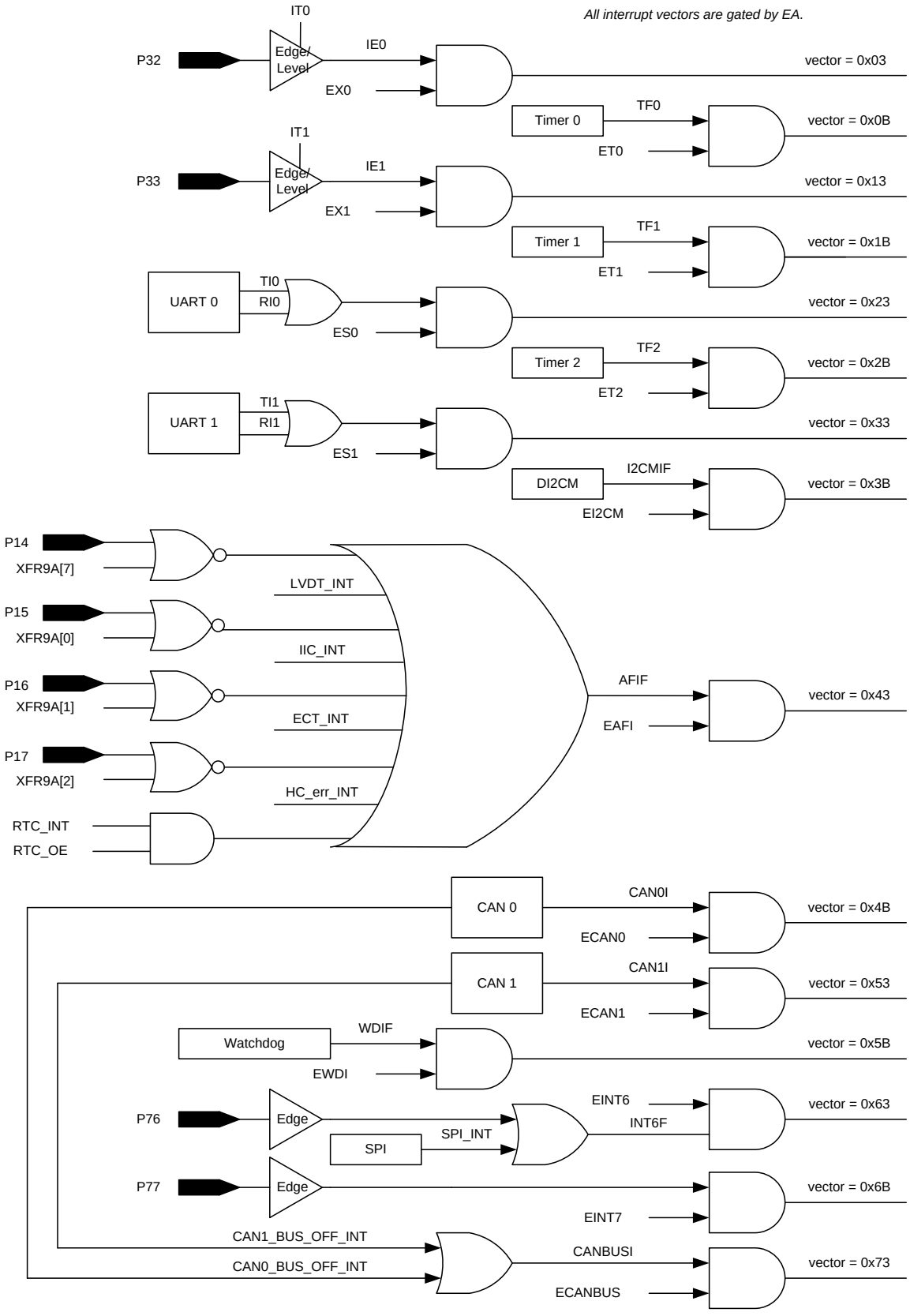


Figure-5. Interrupt Diagram

11. POWER SAVING UNIT

Although the CS8959 is a high-speed micro-controller designed for maximum performance, it also provides power saving unit with two advanced power conservation modes. These modes are Power Management Mode, Low Power Mode and Stop. CS8959 interconnection with clock generator is shown in figure below.

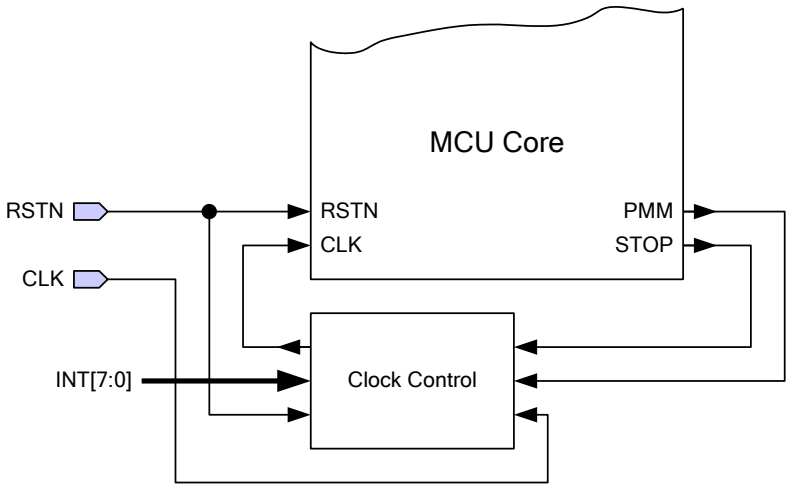


Figure-6. Power Saving Unit Interconnection

The Clock Control block chapter shows example implementation of this module.

11.1 Stop Mode

Stop mode is the lowest power state that the micro-controller can enter. This is achieved by cutting-off frequency provided to CLK pin, resulting in a fully static condition. No processing is possible, timers are stopped, and no serial communication is possible. Processor operation will be postponed on the instruction that sets the STOP bit. Stop mode can be exited in two ways:

1.) A non-clocked interrupt such as the external interrupts INT0-INT7 can be used. Clocked interrupts such as the watchdog timer, internal timers, and serial ports don't operate in Stop mode. Processor operation will resume with the fetching of the interrupt vector associated with the interrupt that caused the exit from Stop mode. When the interrupt service routine will complete, RETI returns the program to the instruction immediately following the one that invoked the Stop mode.

2.) RSTN pin causes the exit from Stop mode. Processor operation will resume execution at address 0X000000.

11.1.1 Stop Bit Power Configuration Register

PCON(w/r) : (SFR 0X87) Stop Bit Power Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SMOD0	SMOD1	-	-	-	-	STOP	-
Default	0	0	-	-	-	-	0	-

STOP - STOP mode bit
=1 → Enter STOP mode.
- Unimplemented bit. Read as 0 or 1

STOP bit indicates true state of core. It means that clock signal must be disconnected from input CLK pin source immediately. After exit from STOP mode it is automatically cleared by hardware, so software always read it as logic zero.

11.2 Power Management Mode

The Power Management Mode (PMM) feature allows software to dynamically match operating frequency and current consumption with the need for processing power. Instead of the full CLK clock speed provided to CS8959 core, power management mode tells external clock generator unit to divide by 256 CLK frequency to conserve power.

The switchback feature allows the micro-controller to almost instantaneously return to full speed mode upon acknowledgment of an external interrupt or a falling edge on a serial port receiver pin. Additionally, micro-controller operating in PMM would normally be unable to sample an incoming serial transmission to properly receive it. The switchback feature allows micro-controller to return to full speed operation in time to receive incoming serial port data and process interrupts with no loss in performance.

11.2.1 Power Management Register

PMR(w/r) : (SFR 0XC4) Power Management Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	CD1	CD0	SWB	-	-	-	-	-
Default	0	0	0	-	-	-	-	-

CD[1:0] = 11 → Enable Power Management Mode (PMM) and clock speed divided by 256
= 01 → Disable Power Management Mode (PMM) and return full-speed mode
SWB - Switchback enable
= 1 → Enable interrupts and serial port causes switchback.
= 0 → Interrupts and serial port don't affect clock CD[1:0]
- Unimplemented bit. Read as 0 or 1.

The CS8959 incorporate a STATUS(0XC5) register to prevent the device from accidentally reducing the clock rate during the servicing of an external interrupt or serial port activity.

11.2.2 Status Register

STATUS(r) : (SFR 0XC5) Status Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	-	HIP	LIP	-	SPTA1	SPRA1	SPTA0	SPRA0
Default	-	0	0	-	0	0	0	0

HIP - High Priority (HP) interrupt status
LIP - Low Priority (LP) interrupt status
SPTA1 - UART1 transmitter activity status
SPRA1 - UART1 receiver activity status
SPTA0 - UART0 transmitter activity status
SPRA0 - UART0 receiver activity status
- - Unimplemented bit. Read as 0 or 1.

This register can be interrogated to determine if a high priority, or low priority interrupt is in progress, or if serial port activity is occurring. Based on this information the software can delay or reject a planned change in the clock divider rate. Table below shows Power Management related registers and bits.

11.2.3 Power Management Related Bit

Bit name	Location	Function	Read/Write access
CD[1:0]	PMR.7-6	Clock Divider control CD[1:0] 0 1 full speed (default after reset) 1 1 divided by 256 (PMM) - 0 not applicable	Unrestricted
SWB	PMR.5	Switchback enable 0- interrupts and serial port don't affect clock CD[1:0] 1- enabled interrupts and serial port causes switchback	Unrestricted
HIP	STATUS.6	High Priority (HP) interrupt status 0- no HP interrupt 1- HP interrupt progressing	Read only
LIP	STATUS.5	Low Priority (LP) interrupt status 0- no LP interrupt 1- LP interrupt progressing	Read only
SPTA1	STATUS.3	UART1 transmitter activity status 0- UART1 transmitter inactive 1- UART1 transmitter active	Read only
SPRA1	STATUS.2	UART1 receiver activity status 0- UART1 receiver inactive 1- UART1 receiver active	Read only
SPTA0	STATUS.1	UART0 transmitter activity status 0- UART0 transmitter inactive 1- UART0 transmitter active	Read only
SPRA0	STATUS.0	UART0 receiver activity status 0- UART0 receiver inactive 1- UART0 receiver active	Read only

11.3 Low Power Mode

When RTC is used, CS8959 will add a more Low Power Mode. Via

```
MOV DPTR, #0F55h
MOV A, #xx01xxxxb
MOVX @DPTR, A
```

System will run at a lower frequency from RTC OSCPAD (RTC_X1). System CLK will be down to 1/2 RTC OSC frequency, and the system OSC will stop. When user wants to run back to full speed, following things will be done:

```
MOV DPTR, #0F55h
MOV A, #xx11xxxxb
MOVX @DPTR, A                    /* Wakeup OSCPAD */
NOP (NOP number =0,1….)
MOV DPTR, #0F55h
MOV A, #xx10Xxxxb
MOVX @DPTR, A                    /* return to full speed */
NOP (NOP number =1, 2…)
```

SEL (r/w) : (XFR 0X0F55) RTC and CLKO select (default 0X20)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	-	TRTC	WKOSC	LPS	CKSL1	CKSL2	-	-
Default	-	0	1	0	0	0	-	-

TRTC = 1 → RTC clock source from internal clock generator
= 0 → RTC clock source from RTC_X1
WKOSC Write 1 to wake up the external clock X1 from Low power mode.
LPS Low power mode select
= 0 → The system clock source is from external clock input X1
= 1 → The system clock source is from RTC_X1
CKSL2, CKSL1 = 00 → CLKO is system clock
= 01 → CLKO is system clock divided 4
= 10 → CLKO is system clock divided 2
= 11 → CLKO is system clock divided 8

11.4 PMM And Peripherals

The Power Management Mode (PMM) reduces power consumption by externally dividing the clock signal provided to the micro-controller, causing it to operate at a reduced speed. When PMM is invoked, the frequency of system clock should be divided by 256. Note that all internal functions, on-board timers (including serial port baud rate generation), watchdog timer, and software timing loops will also run at the reduced speed.

PMM is entered and exited by setting the Clock Rate Divider bits (PMR.7-6). In addition, use of the switchback feature is possible to affect a return from PMM to the full speed mode. This allows both hardware and software to cause an exit from PMM. It is the responsibility of the software to test for UART activity before attempting to change speed, as a modification of the clock divider bits during a UART operation will corrupt the data. In general, it is not possible to generate standard baud rates while in PMM, and the user is advised to avoid PMM or use the switchback feature if UART operation is desired.

11.5 Switchback Feature

The switchback feature solves one of the most vexing dilemmas faced by power conscious systems. Many applications are unable to use Stop mode because they require constant computation. The switchback feature allows a system to operate at a relatively slow speed, and burst to a faster mode when required by an external event.

Enable this feature by setting the SWB bit (PMR.5), a qualified interrupt (interrupt which has occurred and been acknowledged) or serial port reception or transmission cause the micro-controller to return to full speed mode. An interrupt must be enabled and also not blocked by a higher priority interrupt. After the event is completed, software can manually return the micro-controller to PMM. The following sources can trigger switchback:

1. AFI interrupts if enabled.
2. Any external interrupt 0/1/2/3/4/5/6/7 if enabled.
3. UART0/UART1serial start bit detected.
4. UART0/UART1 transmit buffer loaded.
5. Watchdog timer reset.
6. External reset by PAD RSTN.

In the case of a UART-initiated switchback, the switchback is not generated by the associated interrupt. This is because a device operating in PMM will not be able to correctly receive a byte of data to generate an interrupt. Instead, a switchback is generated by a UART reception on the falling edge associated with the start bit, if the associated receiver enable bit (SCON0.4 or SCON1.4) is set. For UART transmissions, a switchback is generated when the UART buffer is loaded. This ensures the micro-controller will be operating in full speed mode when the data is being transmitted, and eliminates the need for a write to the CD[1:0] bits to exit PMM before transmitting. The switchback feature is unaffected by the state of the serial port interrupt flags.

11.6 Switchback Feature Timing

The timing of the switchback is dependent on the source. Interrupt–initiated switchbacks will occur at the start of the first cycle following the event initiating the switchback. If the current instruction in progress is a write to the IE, IP, EXIE, or EXIP registers, interrupt processing will be delayed until the completion of the following instruction. UART-initiated switchbacks occur at the start of the instruction following the MOV that loads SBUF0 or SBUF1. UART-initiated switchbacks occur during the cycle in which the falling edge was detected. There are a few points that must be considered when using a serial port reception to generate a switchback. Under normal circumstances, noise on the line or an aborted transmission would cause the serial port to timeout and the data to be ignored. This presents a problem if the switchback is used, however, because a switchback would occur but there is no indication to the system that one has occurred. If PMM and serial port switchback functions are used in a noisy environment, the user is advised to periodically check if the device has accidentally exited PMM.

A similar problem can occur if multiprocessor communication protocols are used in conjunction with PMM. The problem is that an invalid address that should be ignored by a particular processor will still generate a switchback. As a result it is not recommended to use a multiprocessor communication scheme in conjunction with PMM. If the system power considerations will allow for an occasional erroneous switchback, a polling scheme can be used to place the device back into PMM.

11.7 Pulse Mode

In CS8959, there is another power save option by turning off the embedded flash when not in access period. Users can turn on Pulse Mode by setting PUSEN bit during normal operation mode or Low Power Mode. In order to meet the embedded flash timing requirement, the maximal Oscillator input frequency is 16MHz.

ECCINTEN(w/r) : (XFR 0X0F9C) Error Correction Code interrupt enable. (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	PUSEN	-	-	PRGEN	D256EN	D4KEN	CAN0EN	CAN1EN
Default	0	-	-	0	0	0	0	0

PUSEN = 1 → Enable Pulse Mode.
= 0 → Disable Pulse Mode.

12. TIMERS

The CS8959 contains three 16-bit timer/counters, Timer 0, Timer 1 and Timer 2. In the “timer mode”, Timer 0,1 registers are incremented every 12 or 4 CLK periods when appropriate timer is enabled. In the “timer mode”, Timer 2 registers are incremented every 12 or 2 CLK periods (depends on working mode) when it is enabled. In the “counter mode” the timer registers are incremented every falling transition on theirs corresponding input pins: T0, T1 or T2. The input pins are sampled every CLK period.

12.1 Timer 0 And Timer 1

Timer 0 and Timer 1 are fully compatible with the standard 8051 timers. Each timer consist of two 8-bit registers TH0(0X8C), TL0(0X8A), TH1(0X8D), TL1(0X8B).Timers 0,1 work in the same four modes. The modes are described below.

12.1.1 Timer 0,Timer1 Modes

M1	M0	Mode	Function Description
0	0	0	THx operates as 8-bit timer/counter with a divide by 32 prescaler server by lower 5-bit of TLx.
0	1	1	16-bit timer/counter. THx and TLx are cascaded
1	0	2	TLx operates as 8-bit timer/counter with 8-bit auto-reload by THx
1	1	3	TL0 is configured as 8-bit timer/counter controlled by the standard Timer 0 bits. TH0 is an 8-bit timer controlled by the Timer 1 controls bits. Timer 1 holds its count.

12.1.2 Timer 0,Timer1 Control Mode Register

TMOD (r/w) : (SFR 0X89) Timer 0,Timer1 Control Mode Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Timer1 Control Bits				Timer0 Control Bits			
Name	GATE	CT	M1	M0	GATE	CT	M1	M0
Default	0	0	0	0	0	0	0	0

GATE - Gating control
1 ---Timer x enable while GATEx pin is high and TRx control bit is set.
0 --- Timer x enable while TRx control bit is set.
CT - Counter or timer select bit
1 ---Counter mode
0 --- Timer mode.
M1,0 - Mode select bits

12.1.3 Timer 0,Timer1 Configuration Register

TCON (r/w) : (SFR 0X88) Timer 0,Timer1 Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
Default	0	0	0	0	0	0	0	0

TR0 - Timer 0 run control bit
TR1 - Timer 1 run control bit.
Note : External input GATE0(P3.2), GATE1(P3.3) pins can be programmed to function as a gate to facilitate pulse width measurements.

12.1.4 Clock Control Register

CKCON(r/w) : (SFR 0X8E) Clock Control Register (default 0X03)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	WD1	WD0	T2M	T1M	T0M	MD2	MD1	MD0
Default	0	0	0	0	0	0	1	1

T2M - Timer 2 clock modulus control bit.
1 : Timer2 clock = OSC clock / 4
0 : Timer2 clock = OSC clock / 12
T1M - Timer 1 clock modulus control bit.
1 : Timer1 clock = OSC clock / 4
0 : Timer1 clock = OSC clock / 12
T0M - Timer 0 clock modulus control bit.
1 : Timer0 clock = OSC clock / 4
0 : Timer0 clock = OSC clock / 12

12.2 Timer 0 Operation

12.2.1 Mode 0

In this mode, the Timer0 register is configured as a 13-bit register. As the count rolls over from all 1s to all 0s, Timer0 interrupt flag TF0 is set. The counted input is enabled to the Timer0 when TCON.4=1 and either TMOD.3=0 or GATE0=1. The 13-bit register consists of all 8 bits of TH0 and the lower 5 bits of TL0. The upper 3 bits of TL0 are indeterminate and should be ignored.

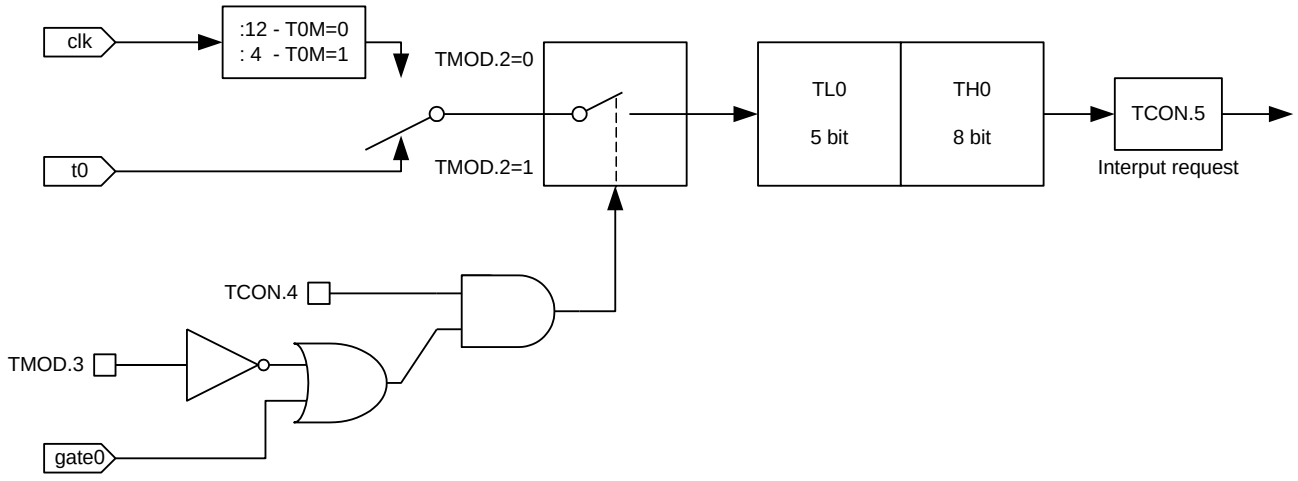


Figure-7. Timer0 Mode 0

12.2.2 Mode 1

Mode 1 is the same as Mode0, except that the timer register is running with all 16 bits.

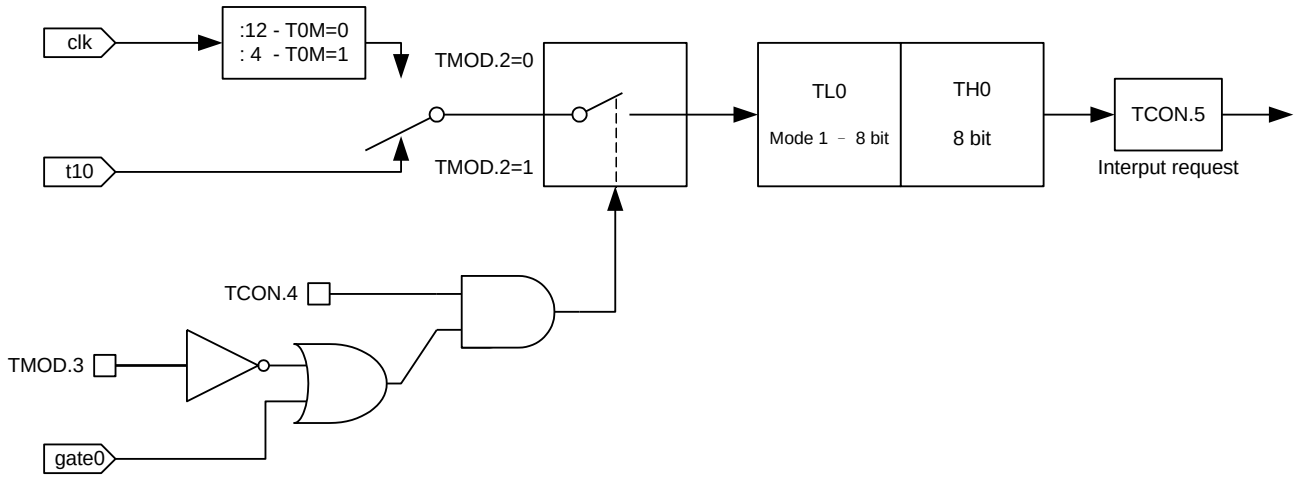


Figure-8. Timer0 Mode 1

12.2.3 Mode 2

Mode 2 configures the timer register as an 8-bit counter(TL0) with automatic reload. Overflow from TL0 not only sets TF0, but also reloads TL0 with the contents of TH0, which is loaded by software. The reload leaves TH0 unchanged.

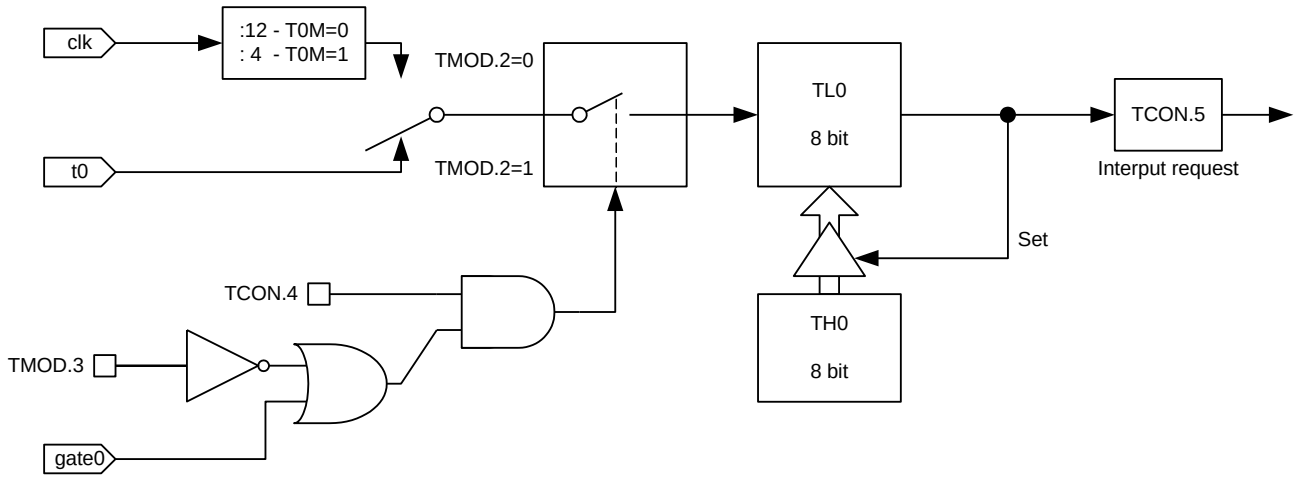


Figure-9. Timer0 Mode 2

12.2.4 Mode 3

Timer0 in Mode 3 establishes TL0 and TH0 as two separate counters. TL0 uses the Timer0 control bits. TH0 is locked into a timer function and uses the TR1 and TF1 flags from Timer1 and controls Timer1 interrupt. Mode 3 provided for applications requiring an extra 8-bit timer/counter. When Timer0 is in Mode 3, Timer1 can be turned on/off by switching it into its own Mode 3, or can still be used by the serial channel as a baud rate generator, or in any application where interrupt from Timer 1 is not required.

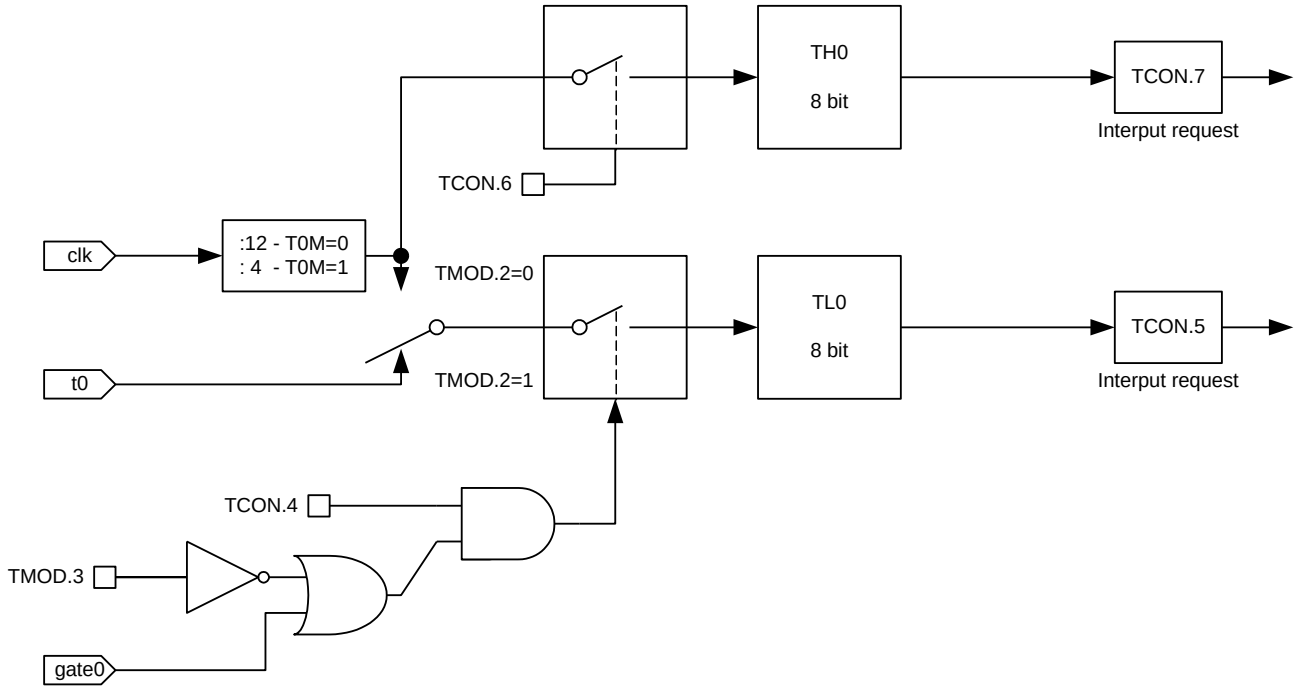


Figure-10. Timer0 Mode 3

12.3 Timer 1 Operation

12.3.1 Mode 0

In this mode, the Timer1 register is configured as a 13-bit register. As the count rolls over from all 1s to all 0s, Timer1 interrupt flag TF1 is set. The counted input is enabled to the Timer1 when TCON.6=1 and either TMOD.7=0 or GATE1=1. The 13-bit register consists of all 8 bits of TH1 and the lower 5 bits of TL1. The upper 3 bits of TL1 are indeterminate and should be ignored.

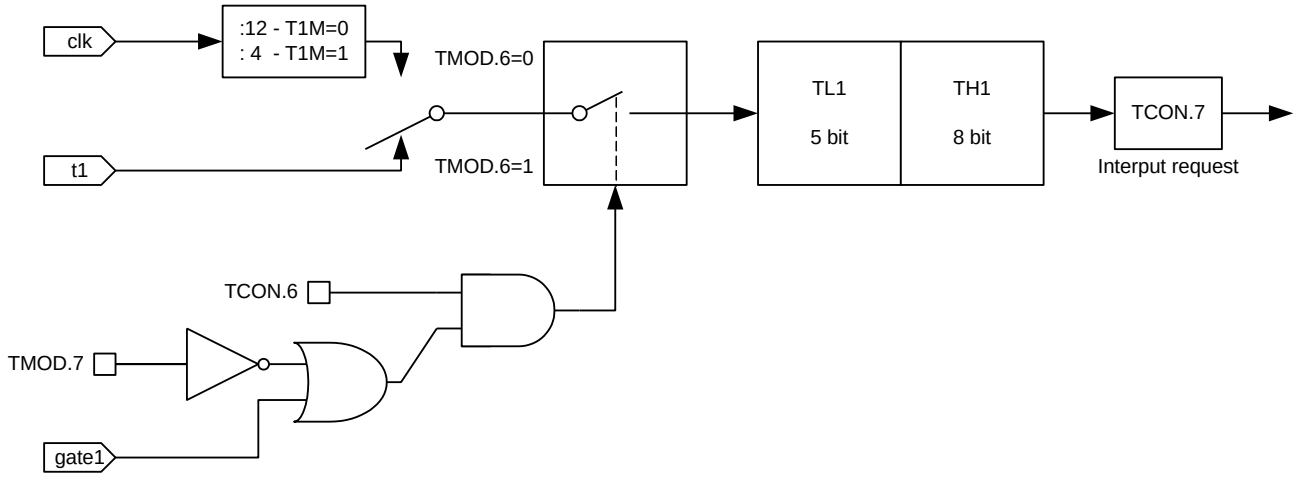


Figure-11. Timer1 Mode 0

12.3.2 Mode 1

Mode 1 is the same as Mode0, except that the timer register is running with all 16 bits.

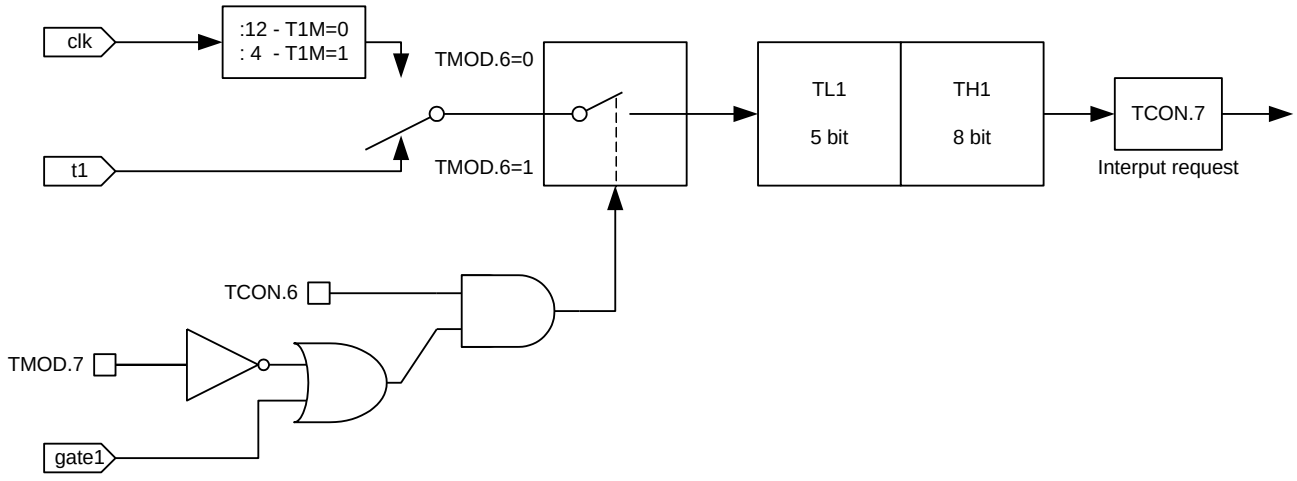


Figure-12. Timer1 Mode1

12.3.3 Mode 2

Mode 2 configures the timer register as an 8-bit counter (TL1) with automatic reload. Overflow from TL1 not only sets TF1, but also reloads TL1 with the contents of TH1, which is loaded by software. The reload leaves TH1 unchanged.

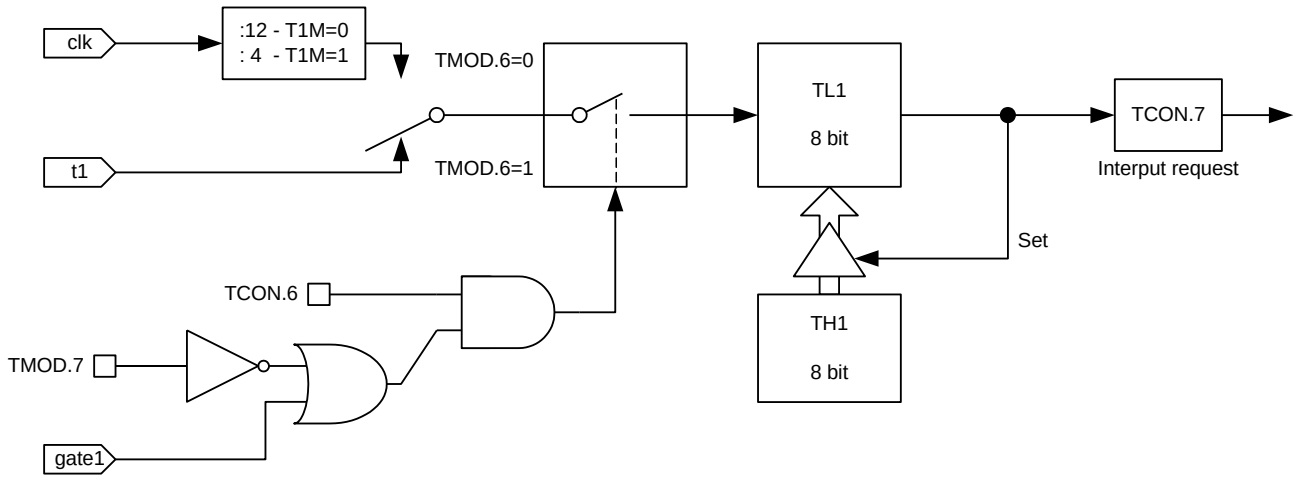


Figure-13. Timer1 Mode1

Mode 3 Timer1 will be held counting. The effect is the same as setting TR1=0.

12.4 Timer 2 Operation

Timer 2 is a 16-bit Timer/Counter, which can operate as either an event timer or an event counter, as selected by C/T2 in the special function register T2CON (0XC8). Timer 2 has three operating modes: Capture, Auto-reload (up or down counting), and Baud Rate Generator, which are selected by bits in the T2CON.

RCLK+TCLK	CP/RL2	TR2	MODE
0	0	1	16-bit Auto-reload
0	1	1	16-bit Capture
1	X	1	Baud rate generator
X	X	0	(off)

12.4.1 Capture Mode

In the capture mode there are two options, which are selected by bit EXEN2 in T2CON. If EXEN2=0, then timer 2 is a 16-bit timer or counter (as selected by C/T2 in T2CON) which, upon overflowing sets bit TF2, the timer 2 overflow bit. This bit can be used to generate an interrupt (by enabling the Timer 2 interrupt bit in the IE register). If EXEN2= 1, Timer 2 operates as described above, but with the added feature that a 1- to -0 transition at external input T2EX causes the current value in the Timer 2 registers, TL2 and TH2, to be captured into registers RCAP2L and RCAP2H, respectively. In addition, the transition at T2EX causes bit EXF2 in T2CON to be set, and EXF2 like TF2 can generate an interrupt (which vectors to the same location as Timer 2 overflow interrupt.

The Timer 2 interrupt service routine can interrogate TF2 and EXF2 to determine which event caused the interrupt). The capture mode is illustrated in Figure-14 (There is no reload value for TL2 and TH2 in this mode. Even when a capture event occurs from T2EX, the counter keeps on counting T2EX pin transitions or osc/12 pulses.).

Timer 2 Configuration Register

T2CON (r/w) : (SFR 0XC8) Timer 2 Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/T2	CP/RL2
Default	0	0	0	0	0	0	0	0

TF2 - Timer 2 interrupt (overflow) flag. Must be cleared by software.
The flag will not be set when either RCLK or TCLK is set.
EXF2 - Falling edge indicator on T2EX pin when EXEN2=1. Must be cleared by software.
RCLK - Receive clock enable
1 - UART0 receiver is clocked by Timer 2 overflow pulses
0 - UART0 receiver is clocked by Timer 1 overflow pulses
TCLK - Transmit clock enable.
1 - UART0 transmitter is clocked by Timer 2 overflow pulses
0 - UART0 transmitter is clocked by Timer 1 overflow pulses
EXEN2 - Enable T2EX pin functionality.
1 - allows capture or reload as a result of T2EX pin falling edge
0 - ignore T2EX events
TR2 - Start/stop Timer 2
1 - start
0 - stop
C/T2 - Timer/counter select
1 - external event counter. Clock source is T2 pin
0 - timer Internally clocked
CP/RL2 - Capture/reload select
1 - T2EX pin falling edge causes capture to occur when EXEN2=1
0 - automatic reload occurs: on Timer 2 overflow or falling edge of T2EX pin when EXEN2=1. When RCLK or TCLK is set this bit is ignored and automatic reload on Timer 2 overflow is forced.

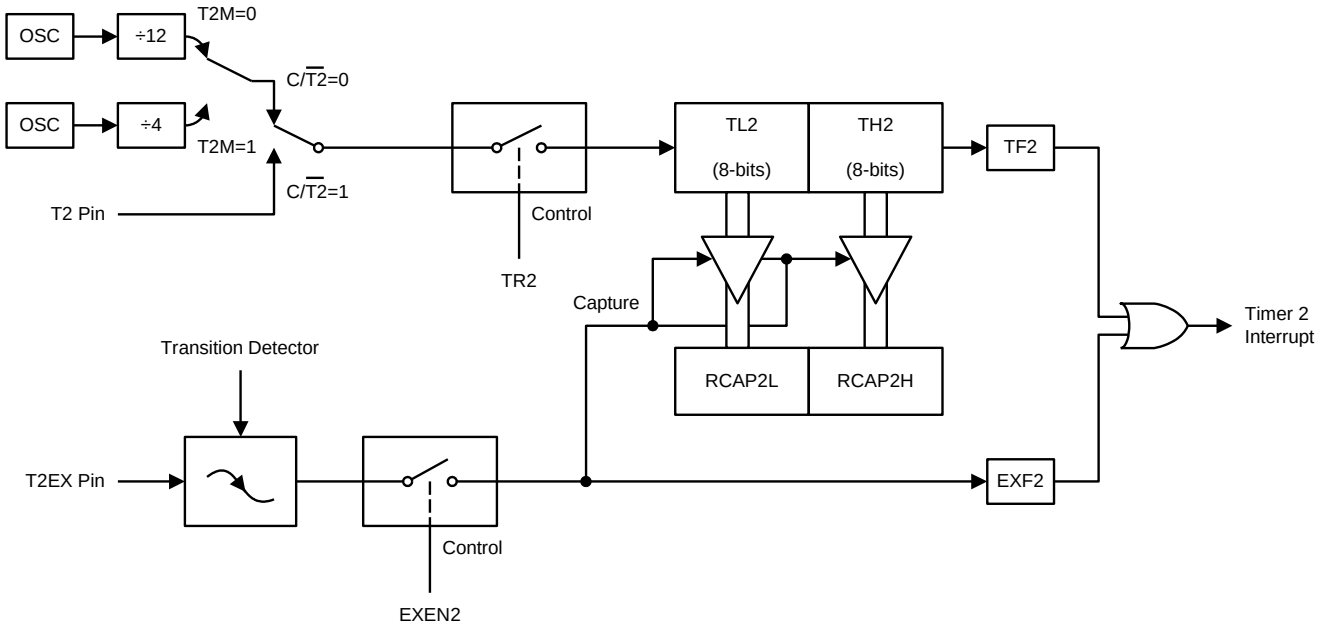


Figure-14. Timer 2 Capture Mode

12.4.2 Auto-Reload Mode (Up or Down Counter)

In the 16-bit auto-reload mode, Timer 2 can be configured (as either a timer or counter (C/T2 in T2CON)) then programmed to count up or down. The counting direction is determined by bit DCEN (Down Counter Enable) which is located in the T2MOD register (0XC9). When reset is applied the DCEN=0 which means Timer 2 will default to counting up. If DCEN bit is set, Timer 2 can count up or down depending on the value of the T2EX pin.

Timer2 T2MOD Register

T2MOD (r/w) : (SFR 0XC9) Timer 2 T2MOD Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	-	-	-	-	-	-	-	DCEN
Default	-	-	-	-	-	-	-	0

Timer 2 which will count up automatically when DCEN=0. In this mode there are two options selected by bit EXEN2 in T2CON register. If EXEN2=0, then Timer 2 counts up to 0FFFFH and sets the TF2 (Overflow Flag) bit upon overflow. This causes the Timer 2 registers to be reloaded with the 16-bit value in RCAP2L and RCAP2H. The values in RCAP2L and RCAP2H are preset by software means. If EXEN2=1, then a 16-bit reload can be triggered either by an overflow or by a 1-to-0 transition at input T2EX. This transition also sets the EXF2 bit. The Timer 2 interrupt, if enabled, can be generated when either TF2 or EXF2 are 1.

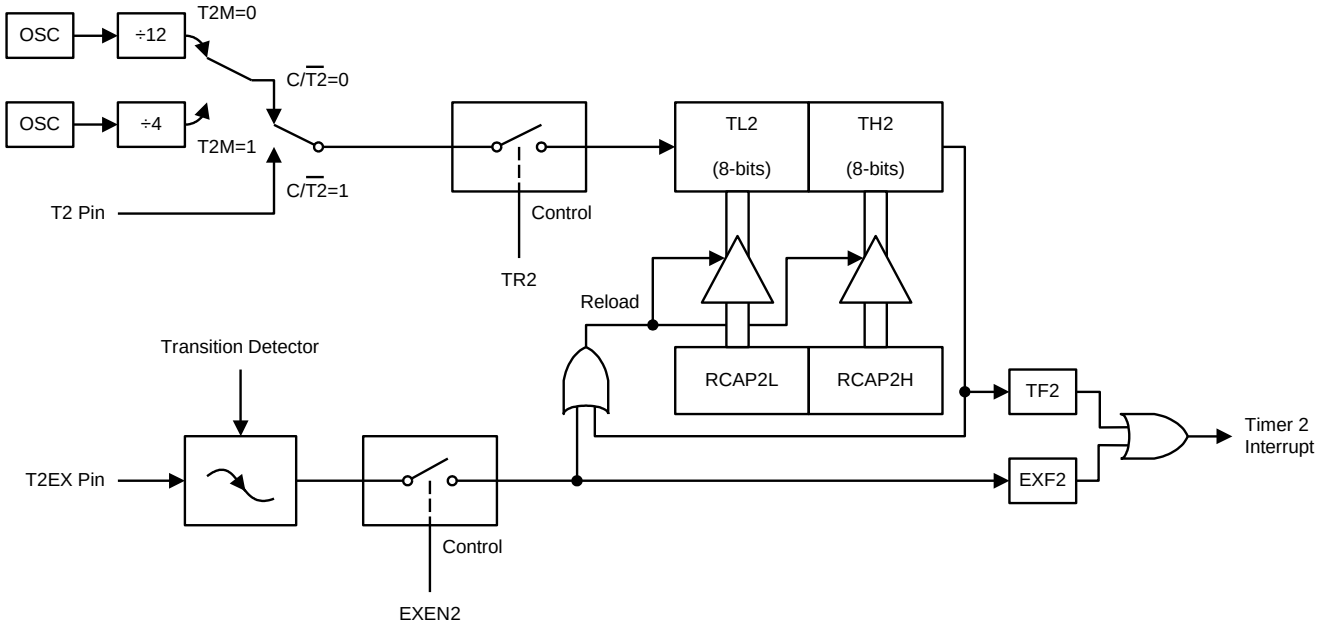


Figure-15. Timer 2 Auto-reloaded Mode (DCEN =0)

DCEN=1 which enables Timer 2 to count up or down. This mode allows pin T2EX to control the direction of count. When a logic 1 is applied at pin T2EX Timer 2 will count up. Timer 2 will overflow at 0FFFFH and set the TF2 flag, which can then generate an interrupt, if the interrupt is enabled. This timer overflow also causes the 16–bit value in RCAP2L and RCAP2H to be reloaded into the timer registers TL2 and TH2.

When a logic 0 is applied at pin T2EX this causes Timer 2 to count down. The timer will underflow when TL2 and TH2 become equal to the value stored in RCAP2L and RCAP2H. Timer 2 underflow sets the TF2 flag and causes 0FFFFH to be reloaded into the timer registers TL2 and TH2.

The external flag EXF2 toggles when Timer 2 underflows or overflows. This EXF2 bit can be used as a 17th bit of resolution if needed. The EXF2 flag does not generate an interrupt in this mode of operation.

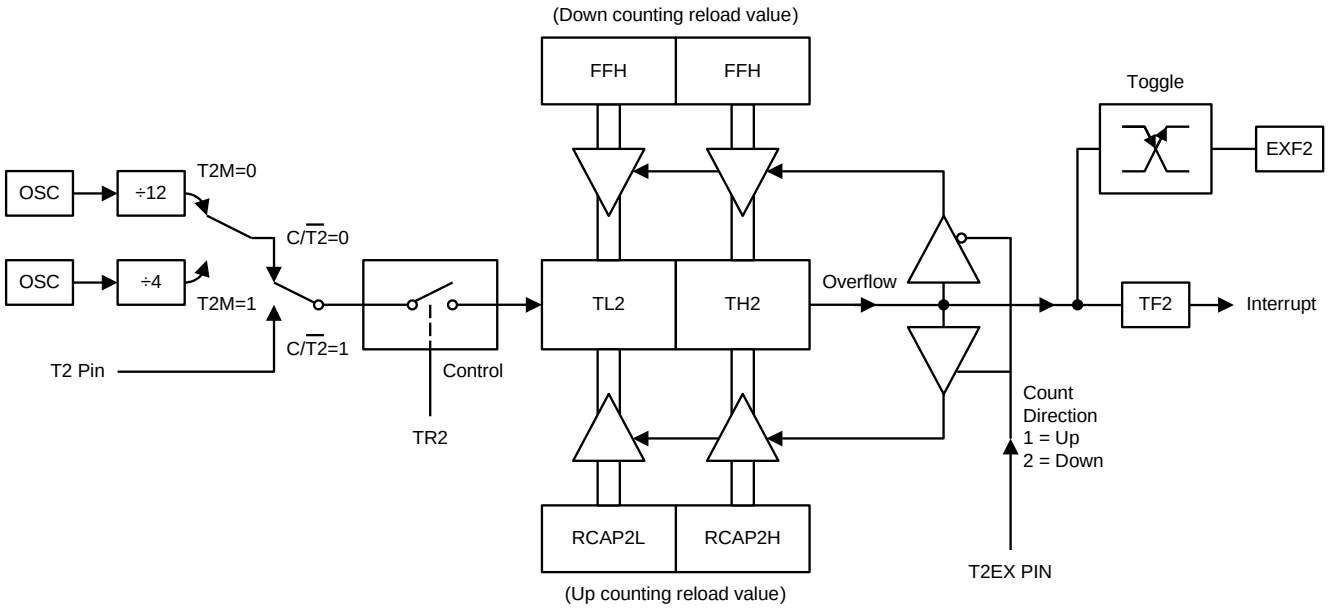


Figure-16. Timer 2 Auto-Reloaded Mode (DCEN =1)

12.4.3 Baud Rate Generator Mode

Bits TCLK and/or RCLK in T2CON allow the serial port transmit and receive baud rates to be derived from either Timer 1 or Timer 2. When TCLK= 0, Timer 1 is used as the serial port (UART0) transmit baud rate generator. When TCLK= 1, Timer 2 is used as the serial port (UART0) transmit baud rate generator. RCLK has the same effect for the serial port receive baud rate. With these two bits, the serial port can have different receive and transmit baud rates – one generated by Timer 1, the other by Timer 2.

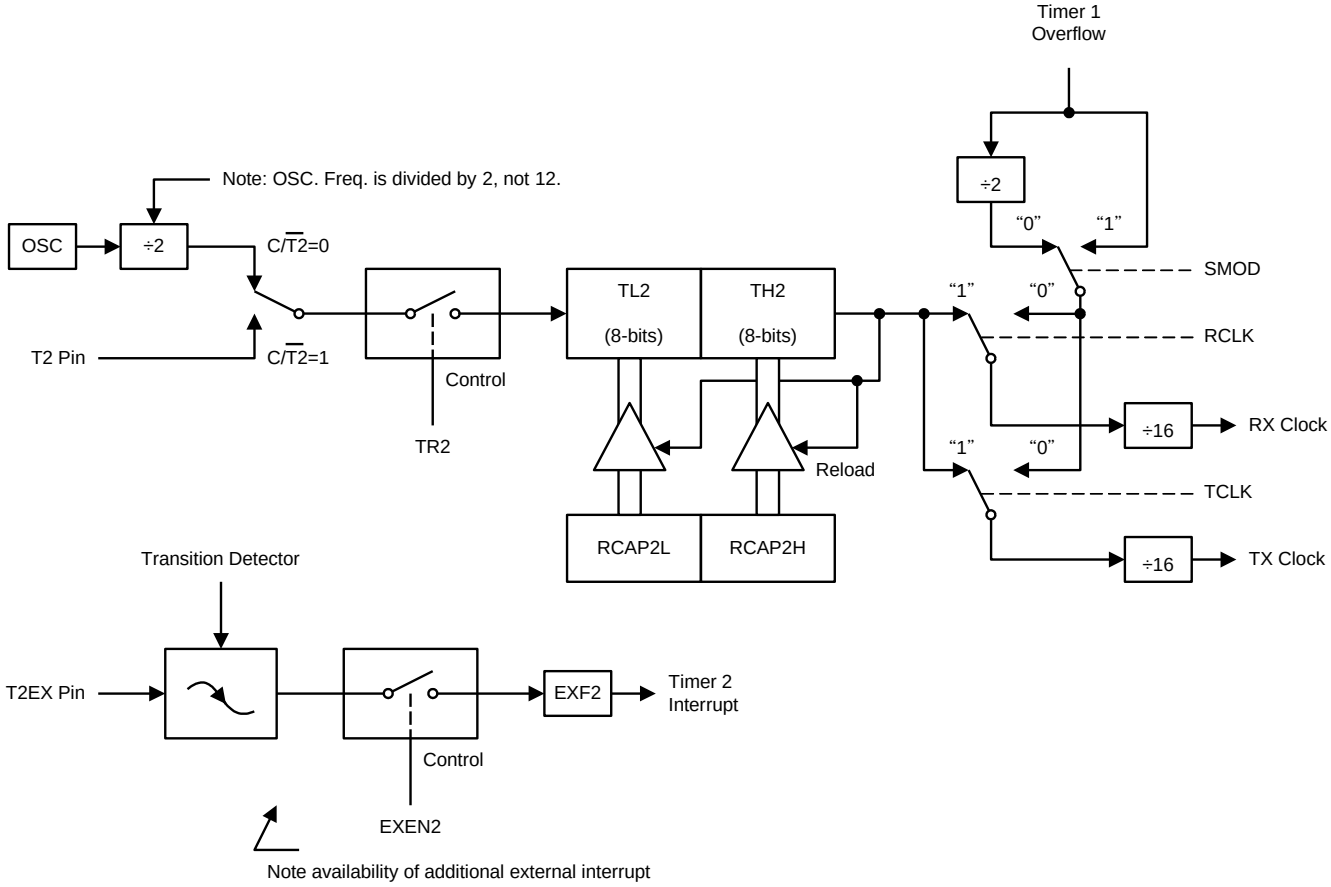


Figure-17. Timer 2 As A Baud Rate Generator Mode

The Timer 2 as a baud rate generator mode is valid only if RCLK and/or TCLK = 1 in T2CON register. Note that a rollover in TH2 does not set TF2, and will not generate an interrupt. Thus, the Timer 2 interrupt does not have to be disabled when Timer 2 is in the baud rate generator mode. Also if the EXEN2 (T2 external enable flag) is set, a 1-to-0 transition in T2EX (Timer/counter 2 trigger input) will set EXF2 (T2 external flag) but will not cause a reload from (RCAP2H, RCAP2L) to (TH2,TL2). Therefore when Timer 2 is in use as a baud rate generator, T2EX can be used as an additional external interrupt, if needed. When Timer 2 is in the baud rate generator mode, one should not try to read or write TH2 and TL2.

As a baud rate generator, Timer 2 is incremented every state time (osc/2) or asynchronously from pin T2; under these conditions, a read or write of TH2 or TL2 may not be accurate. The RCAP2 registers may be read, but should not be written to, because a write might overlap a reload and cause write and/or reload errors. The timer should be turned off (clear TR2) before accessing the Timer 2 or RCAP2 registers.

13. SERIAL PORTS (UARTS)

Both serial ports are full duplex, meaning it can transmit and receive concurrently. They are receive double-buffered, meaning they can commence reception of a second byte before a previously received byte has been read from the receive register. Writing to SBUFx loads the transmit register, and reading SBUFx reads a physically separate receive register. The serial ports can operate in 4 modes: one synchronous and three asynchronous modes.

Mode 2 and 3 have a special provision for multiprocessor communications. This feature is enabled by setting SMx2 bit in SCONx register. The master processor first sends out an address byte which identifies the target slave. An address byte differs from a data byte in that the 9th bit is 1 in an address byte and 0 in a data byte. With SMx2 = 1, no slave will be interrupted by a data byte. An address byte, will interrupt all slaves. The addressed slave will clear its SMx2 bit and prepare to receive the data bytes that will be came. The slaves that weren't being addressed leave their SMx2 set and ignoring the incoming data.

13.1 UART0

The UART0 has the same functionality as a standard 8052 UART. The UART0 related registers are: SBUF0(0X99), SCON0(0X98), PCON(0X87), IE(0XA8) and IP(0XB8). The UART0 data buffer (SBUF0) consists of two separate registers: transmit and receive registers. A data writes into the SBUF0 sets this data in UART0 output register and starts a transmission. A data reads from SBUF0, reads data from the UART0 receive register.

UART0 Buffer Register

SBUF0 (r/w) : (SFR 0X99) UART0 Buffer Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SB0.7	SB0.6	SB0.5	SB0.4	SB0.3	SB0.2	SB0.1	SB0.0
Default	0	0	0	0	0	0	0	0

UART0 Configuration Register

SCON0 (r/w) : (SFR 0X98) UART0 Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SM00	SM01	SM02	REN0	TB08	RB08	TI0	RI0
Default	0	0	0	0	0	0	0	0

SM02 - Enables a multiprocessor communication feature
SM01 - Sets baud rate
SM00 - Sets baud rate
REN0 - If set, enables serial reception. Cleared by software to disable reception.
TB08 - The 9th transmitted data bit in Modes 2 and 3. Set or cleared by the CPU, depending on the function it performs (parity check, multiprocessor communication etc.)
RB08 - In Modes 2 and 3 it is the 9th data bit received. In Mode 1, if SM12 is 0,RB08 is the stop bit. In Mode 0 this bit is not used.
TI0 - Transmit interrupt flag, set by hardware after completion of a serial transfer. Must be cleared by software.
RI0 - Receive interrupt flag, set by hardware after completion of a serial reception. Must be cleared by software

The UART0 modes are presented in table below.

SM00	SM01	Mode	Description	Baud Rate
0	0	0	Shift register	fosc/12
0	1	1	8-bit UART	variable
1	0	2	9-bit UART	fosc/32 or fosc/64
1	1	3	9-bit UART	variable

The UART0 baud rates are presented in the table below.

Mode	Baud Rate
Mode 0	$F_{CLK}/12$
Mode 1, 3	Timer 1 or Timer 2 overflow rate
Mode 2	SMOD0 = 0 $f_{CLK}/64$ SMOD0 = 1 $f_{CLK}/32$

The SMOD0 bit is located in PCON register.

UART0 Bits In Power Configuration Register

PCON0 (r/w) : (SFR 0X98) UART0 Bits In Power Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SMOD0	SMOD1	-	-	-	-	STOP	-
Default	0	0	-	-	-	-	0	-

SMOD0 - UART0 double baud rate bit
- Unimplemented bit. Read as 0 or 1.

URAT0 Bits In Interrupt Enable Register

IE (r/w) : (SFR 0XA8) URAT0 Bits In Interrupt Enable Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	EA	ES1	ET2-	ES0	ET1	EX1	ET0	EX0
Default	0	0	0	0	0	0	0	0

ES0 - RI0&TI0 interrupt enable flag
1 – Enable UART0 RI0 & TI0 interrupt.
0 – Disable UART0 RI0 & TI0 interrupt.

URAT0 Bits In Interrupt Priority Register

IP (r/w) : (SFR 0XB8) URAT0 Bits In Interrupt Priority Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	-	PS1	PT2	PS0	PT1	PX	ET0	EX0
Default	-	0	0	0	0	0	0	0

PS0 - RI0&TI0 interrupt priority flag
1 – Set to high priority
0 – Set to low priority

- Unimplemented bit. Read as 0 or 1.

13.1.1 MODE 0 , SYNCHRONOUS

Pin P30/RX0 serves as bi-direction pad. P31/TX0 output is a shift clock. The baud rate is fixed at 1/12 of the CLK clock frequency. Eight bits are transmitted with LSB first. Reception is initialized by setting the flags in SCON0 as follows: RI0=0 and REN0=1.

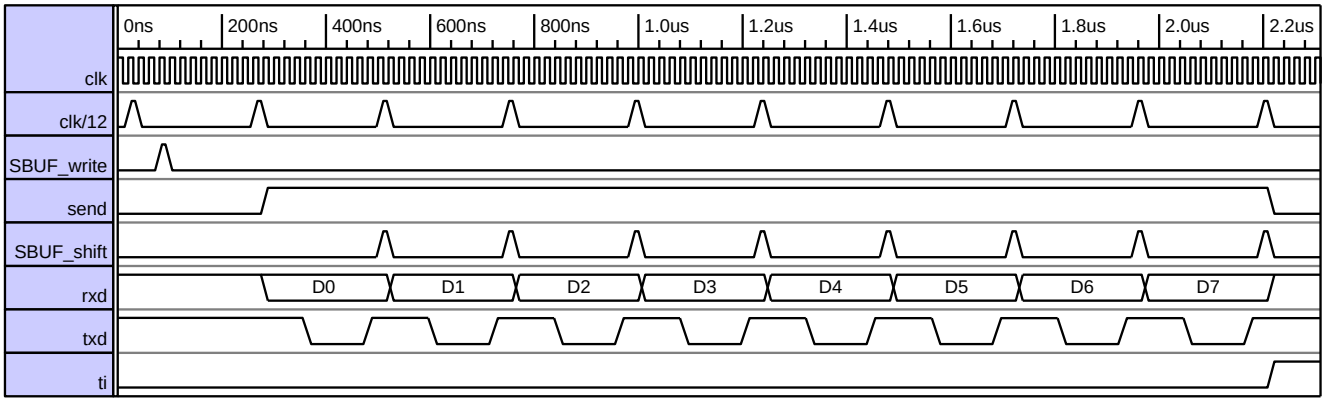


Figure-18. UART0 Transmission Mode 0 Timing Diagram

13.1.2 MODE 1, 8-BIT UART, VARIABLE BAUD RATE, TIMER 1 OR 2 CLOCK SOURCES

Pin P30/RX0 serves as input, and P31/TX0 serves as serial output. 10 bits are transmitted: a start bit (always 0), 8 data bits (LSB first), and a stop bit (always 1). On receive, a start bit synchronizes the transmission, 8 data bits are available by reading SBUF0, and stop bit sets the flag RB08 in the SFR SCON0. The baud rate is variable and depends from Timer 1 or Timer 2 mode. To enable Timer 2 clocking set the TCLK, RCLK bits located in T2CON (0XC8) register.

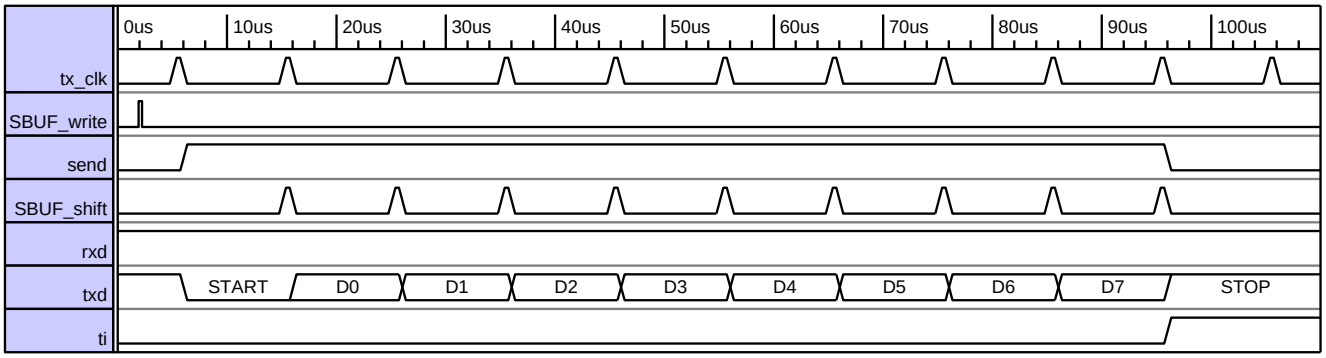


Figure-19. UART0 Transmission Mode 1 Timing Diagram

13.1.3 MODE 2, 9-BIT UART, FIXED BAUD RATE

This mode is similar to Mode 1 with two differences. The baud rate is fixed at 1/32 or 1/64 of system clock frequency, and 11 bits are transmitted or received: a start bit (0), 8 data bits (LSB first), a programmable 9th bit, and a stop bit (1). The 9th bit can be used to control the parity of the UART interface: at transmission, bit TB08 in SCON0 is output as the 9th bit, and at receive, the 9th bit affects RB08 in SCON0.

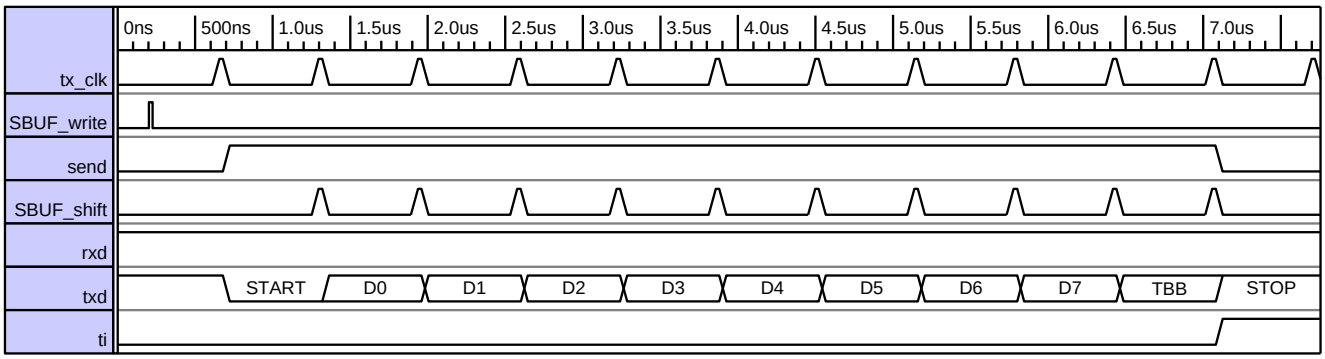


Figure-20. UART0 Transmission Mode 2 Timing Diagram

13.1.4 MODE 3, 9-BIT UART, VARIABLE BAUD RATE, TIMER 1 OR 2 CLOCK SOURCES

The only difference between Mode 2 and Mode 3 is that the baud rate is a variable in Mode 3. When REN0=1 data receiving is enabled. The baud rate is variable and depends from Timer 1 or Timer 2 mode. To enable Timer 2 clocking set the TCLK, RCLK bits located in T2CON (0XC8) register.

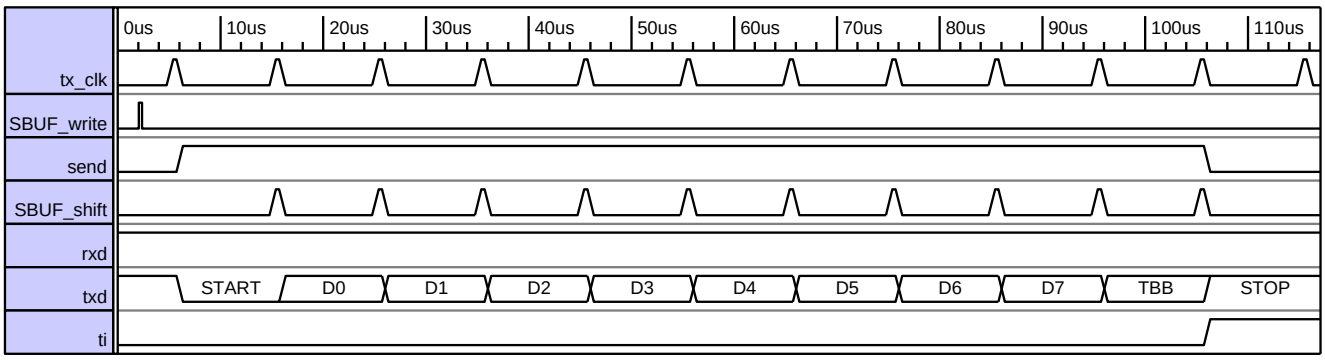


Figure-21. UART0 Transmission Mode 2 Timing Diagram

13.2 UART1

The UART1 has the same functionality as a standard 8052 UART. The UART1 related registers are: SBUF1(0XC1), SCON1(0XC0), PCON(0X87), IE(0XA8) and IP(0XB8). The UART1 data buffer (SBUF1) consists of two separate registers: transmit and receive registers. Data write into the SBUF1 sets this data in UART1 output register and starts a transmission. Data read from SBUF1, reads data from the UART1 receive register.

UART1 Buffer Register

SBUF1 (r/w) : (SFR 0XC1) UART1 Buffer Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SB1.7	SB1.6	SB1.5	SB1.4	SB1.3	SB1.2	SB1.1	SB1.0
Default	0	0	0	0	0	0	0	0

UART1 Configuration Register

SCON1 (r/w) : (SFR 0XC0) UART1 Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SM10	SM11	SM12	REN1	TB18	RB18	TI1	RI1
Default	0	0	0	0	0	0	0	0

SM12 - Enables a multiprocessor communication feature
SM11 - Sets baud rate
SM10 - Sets baud rate
REN1 - If set, enables serial reception. Cleared by software to disable reception.
TB18 - The 9th transmitted data bit in Modes 2 and 3. Set or cleared by the CPU, depending on the function it performs (parity check, multiprocessor communication etc.)
RB18 - In Modes 2 and 3 it is the 9th data bit received. In Mode 1, if SM12 is 0,RB18 is the stop bit. In Mode 0 this bit is not used.
TI1 - Transmit interrupt flag, set by hardware after completion of a serial transfer. Must be cleared by software.
RI1 - Receive interrupt flag, set by hardware after completion of a serial reception. Must be cleared by software

The UART1 modes are presented in table below.

SM10	SM11	Mode	Description	Baud Rate
0	0	0	Shift register	$F_{CLK}/12$
0	1	1	8-bit UART	variable
1	0	2	9-bit UART	$F_{CLK}/32$ or $F_{CLK}/64$
1	1	3	9-bit UART	variable

The UART1 baud rates are presented in the table below.

Mode	Baud Rate
Mode 0	$F_{CLK}/12$
Mode 1, 3	Timer 1 overflow rate
Mode 2	SMOD = 0 $F_{CLK}/64$ SMOD = 1 $F_{CLK}/32$

The SMOD1 bit is located in PCON register.

UART1 Bits In Power Configuration Register

PCON (r/w) : (SFR 0X87) UART1 Bits In Power Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SMOD0	SMOD1	-	-	-	-	STOP	-
Default	0	0	-	-	-	-	0	-

SMOD1 - UART1 double baud rate bit
- Unimplemented bit. Read as 0 or 1.

URAT1 Bits In Interrupt Enable Register

IE (r/w) : (SFR 0XA8) URAT1 Bits In Interrupt Enable Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	EA	ES1	ET2-	ES0	ET1	EX1	ET0	EX0
Default	0	0	0	0	0	0	0	0

ES1 – RI1&TI1 interrupt enable flag
1 – Enable UART1 RI1 & TI1 interrupt.
0 – Disable UART1 RI1& TI1 interrupt.

URAT1 Bits In Interrupt Priority Register

IP (r/w) : (SFR 0XB8) URAT1 Bits In Interrupt Priority Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	-	PS1	PT2	PS0	PT1	PX	ET0	EX0
Default	-	0	0	0	0	0	0	0

PS1 - RI1&TI1 interrupt priority flag
1 – Set to high priority
0 – Set to low priority

- Unimplemented bit. Read as 0 or 1

13.2.1 MODE 0, SYNCHRONOUS

Pin P12/RX1 serves as bi-direction pad. P13/TX1 output is a shift clock. The baud rate is fixed at 1/12 of the system clock frequency. Eight bits are transmitted with LSB first. Reception is initialized by setting the flags in SCON1 as follows: RI1=0 and REN1=1.

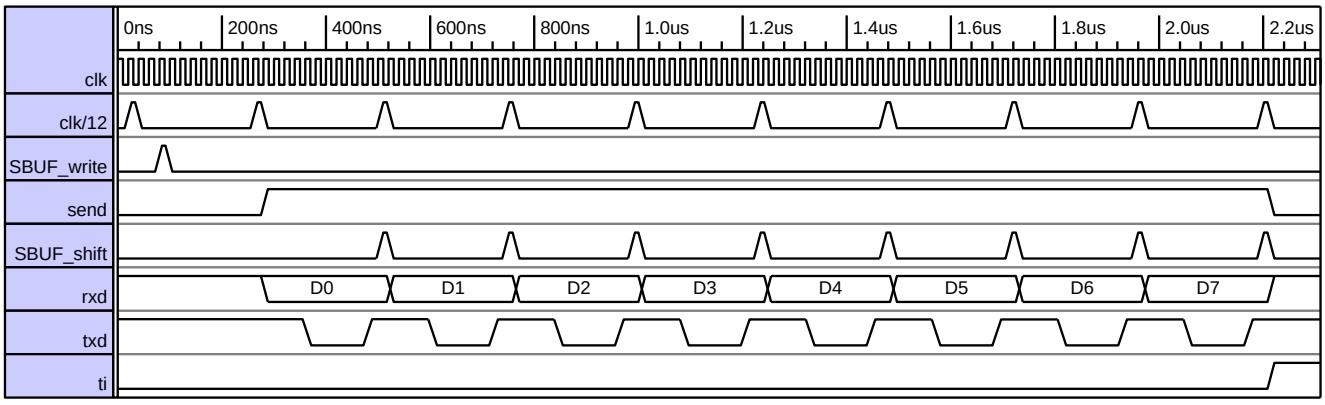


Figure-22. UART1 Transmission Mode 0 Timing Diagram

13.2.2 MODE 1, 8-BIT UART, VARIABLE BAUD RATE, TIMER 1 CLOCK SOURCE

Pin P12/RX1 serves as input, and P13/TX1 serves as serial output. 10 bits are transmitted: a start bit (always 0), 8 data bits (LSB first), and a stop bit (always 1). On receive, a start bit synchronizes the transmission, 8 data bits are available by reading SBUF1, and stop bit sets the flag RB18 in the SFR SCON1. The baud rate is variable and depends from Timer 1 only.

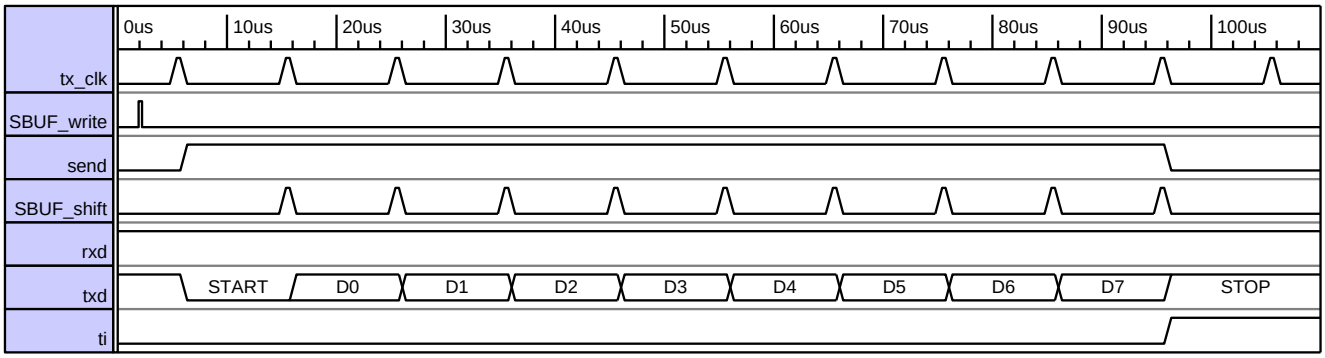


Figure-23. UART1 Transmission Mode 1 Timing Diagram

13.2.3 MODE 2, 9-BIT UART, FIXED BAUD RATE

This mode is similar to Mode 1 with two differences. The baud rate is fixed at 1/32 or 1/64 of system clock frequency, and 11 bits are transmitted or received: a start bit (0), 8 data bits (LSB first), a programmable 9th bit, and a stop bit (1). The 9th bit can be used to control the parity of the UART interface: at transmission, bit TB18 in SCON1 is output as the 9th bit, and at receive, the 9th bit affects RB18 in SCON1.

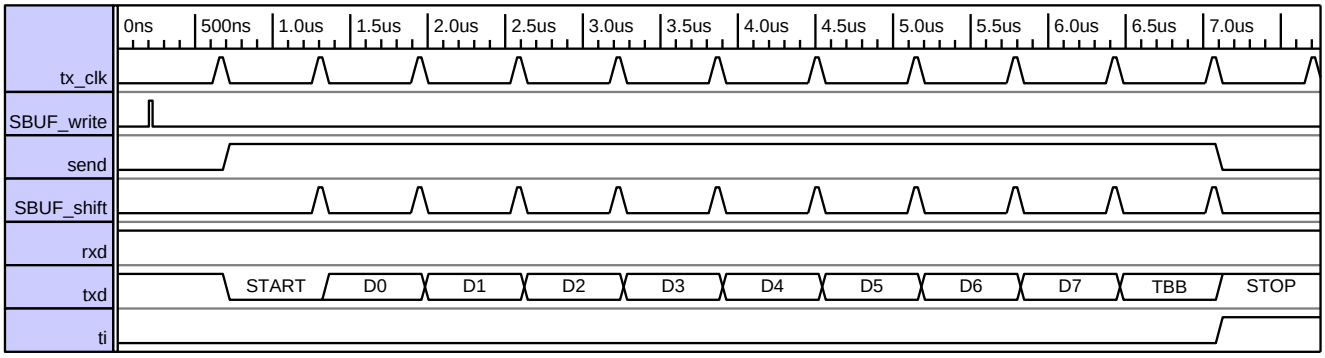


Figure-24. UART1 Transmission Mode 2 Timing Diagram

13.2.4 MODE 3, 9-BIT UART, VARIABLE BAUD RATE, TIMER 1 CLOCK SOURCE

The only difference between Mode 2 and Mode 3 is that the baud rate is a variable in Mode 3. When REN1=1 data receiving is enabled. The baud rate is variable and depends from Timer 1 only.

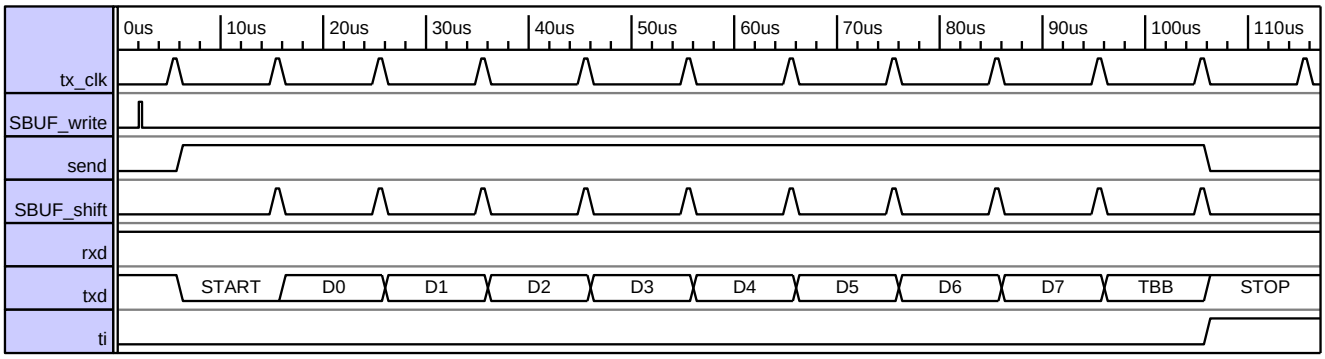


Figure-25. UART1 Transmission Mode 3 Timing Diagram

14. MDU – MULTIPLICATION AND DIVISION UNIT

This arithmetic unit (called MDU) provides fast 32-bit division, 16-bit multiplication and shift and normalize features. All operations are unsigned integer operations. The MDU is handled by seven registers MD0 to MD5, contain the operands and the result and one control register called ARCON.

MDU Registers Location

SFR Name	Address	Description
MD0	0XF9	MDU data register 0
MD1	0XFA	MDU data register 1
MD2	0XFB	MDU data register 2
MD3	0XFC	MDU data register 3
MD4	0XFD	MDU data register 4
MD5	0XFE	MDU data register 5
ARCON	0XFF	MDU control register

ARCON Registers

ARCON (r/w) : (SFR 0XFF) ARCON Registers (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	MDEF	MDOV	SLR	SC.4	SC.3	SC.2	SC.1	SC.0
Default	0	0	0	0	0	0	0	0

MDEF - Error flag. Indicates an improperly performed operation. Set by hardware when an operation is restarted by a write access to MDx before the first operation has been completed. MDEF is automatically cleared after being read.

MDOV - Overflow flag. Controlled by hardware only. Set if division by zero or multiplication result is greater than 0X0000FFFFh

SLR - Shift direction bit
1 – shift right
0 – shift left

SC[4:0] - Shift counter bits. When preset with 00000B, normalizing is selected. After operation SC[4:0] contain the number of normalizing shifts performed. When set with a value differ than 0, shift operation is started. The number of shifts performed is determined by the count written to SC[4:0]

Table below describes the four general operations the MDU is able to perform:

MDU Operations Timing

Operation	Result	Reminder	Execution time [CLK]
32b/16b	32b	16b	17
16b/16b	16b	16b	9
16bx16b	32b	-	10
32b normalize	-	-	3-20
32b shift left/right	-	-	3-18

MDU operation is consisted of three phases:
1. Loading MDx registers in an appropriate order dependent on operation.
2. Executing operation.
3. Result reading from MDx registers.

MDU Read/Write Sequences

Operation	Shift/Norm		32b/32b			16b/16b			16bx16b		
First write	MD0	Low	MD0	Dividend	Low	MD0	Dividend	Low	MD0	Multiplicand	Low
	MD1		MD1	Dividend		MD1	Dividend	High	MD4	Multiplicator	Low
	MD2		MD2	Dividend							
	MD3	High	MD3	Dividend	High						
			MD4	Divisor	Low	MD4	Divisor	Low	MD1	Multiplicand	High
Last write	ARCON		MD5	Divisor	High	MD5	Divisor	High	MD5	Multiplicator	High
First read	MD0	Low	MD0	Quotient	Low	MD0	Quotient	Low	MD0		Low
	MD1		MD1	Quotient		MD1	Quotient	High	MD1		
	MD2		MD2	Quotient					MD2	Product	
	MD3	High	MD3	Quotient	High				MD3		High
			MD4	Reminder	Low	MD4	Reminder	Low			
Last read			MD5	Reminder	High	MD5	Reminder	High			

14.1 Division And Multiplication Operations

Below are described sequential operations performed to execute division and multiplication operations on required data arguments.

14.1.1 WRITE SEQUENCE

The first and the last write operations in phase one are fixed for every calculation of the MDU. All write operations in-between determine the type of MDU calculation. A write to MD0 is the first transfer to be done in any case. This write resets the MDU and triggers the error flag mechanism. The next two or three write operations select the calculation type. The last write-to-MD5 finally starts selected MUL/DIV operation.

14.1.2 READ SEQUENCE

Any read-out of the MDx registers should begin with MD0. The last read from MD5 (division) or MD3 (multiplication) determines the end of a whole calculation and releases the error flag mechanism. There is no restriction on the time within which a calculation must be completed. Any new write access to MD0 starts a new calculation, no matter whether the read-out of the former result has been completed or not.

14.2 Normalize Operation

Normalizing is done on an integer variable stored in MD0 to MD3. “To normalize” means, that all reading zeroes of an integer variable in registers MD0 to MD3 are removed by shift left operations. The whole operation is completed when the MSB of number contains a logic 1.

To select a normalize operation, the five bit field ARCON.0 to ARCON.4 must be cleared. That means, a write-to-ARCON instruction with the value XXX0 0000B starts the operation. After normalizing, bits ARCON.0 to ARCON.4 contain the number of shift left operations which were done. This number may further on be used as an exponent.

Register ARCON controls an up to 32-bit wide normalize and shift operation in registers MD0 to MD3. It also contains the overflow flag and the error flag.

14.2.1 WRITE SEQUENCE

A write-to-MD0 is the first transfer to be done for normalize and shift. This write resets the MDU and triggers the error flag mechanism. To start a shift or normalize operation the last write must access register ARCON.

14.2.2 READ SEQUENCE

The order in which the first three registers MD0 to MD2 are read is not critical. The last read from MD3 determines the end of a whole shift or normalize procedure and releases the error flag mechanism.

Note: Any write access to ARCON triggers a shift or normalize operation and therefore changes the contents of registers MD0 to MD3.

Completion of operation, can also be controlled by the error flag mechanism. The error flag is set if one of the relevant registers (MD0 through MD3) is accessed before the previously commenced operation has been completed. For proper operation of the error flag mechanism, it is necessary to take care that the right write or read sequence to or from registers MD0 to MD3 is maintained.

14.3 Shift Operation

By a write-to-ARCON instruction - a shift left/right operation can be started. In this case register bit SLR (ARCON.5) has to contain the shift direction, and ARCON.0 to ARCON.4 the shift count (which must not be 0). During shift, zeroes come into the left or right end of the registers MD0 or MD3, respectively.

Register ARCON controls an up to 32-bit wide normalize and shift operation in registers MD0 to MD3. It also contains the overflow flag and the error flag.

14.3.1 WRITE SEQUENCE

A write-to-MD0 is the first transfer to be done for normalize and shift. This write resets the MDU and triggers the error flag mechanism. To start a shift or normalize operation the last write must access register ARCON.

14.3.2 READ SEQUENCE

The order in which the first three registers MD0 to MD2 are read is not critical. The last read from MD3 determines the end of a whole shift or normalize procedure and releases the error flag mechanism.

Note: Any write access to ARCON triggers a shift or normalize operation and therefore changes the contents of registers MD0 to MD3!

Completion of operation, can also be controlled by the error flag mechanism. The error flag is set if one of the relevant registers (MD0 through MD3) is accessed before the previously commenced operation has been completed. For proper operation of the error flag mechanism, it is necessary to take care that the right write or read sequence to or from registers MD0 to MD3 is maintained.

14.4 MDU Error Flag

The error fag indicates an improperly performed operation. The error mechanism flag is automatically enabled during MD0 write and disabled during result reading from MD3 (multiplication) or MD5 (division) register.
The flag is set if:
- current calculation has been interrupted or restarted
- mechanism is enabled and read operation is performed

The flag is cleared if: operation is successfully completed and read access is performed

14.5 MDU Overflow Flag

The flag is set if:
- division by zero occurred
- multiplication result is greater than 0X0000FFFF
- normalize operation has been performed on normalized number (MD3(7)=1)

This flag is read only.

15. DI2CM - MASTER IIC BUS CONTROLLER

15.1 The I2C-Bus Short Specification

15.1.1 THE I2C-BUS CONCEPT

The I2C-bus supports any IC fabrication process (NMOS, CMOS, bipolar). Two wires, serial data (SDA) and serial clock (SCL), carry information between the devices connected to the bus. Each devices is recognized by a unique address – whether it's a micro-controller, LCD driver, memory or keyboard interface – and can operate as either a transmitter or receiver, depending on the function of the device. Obviously an LCD driver is only a receiver, whereas a memory can both receive and transmit data. In addition to transmitters and receivers, devices can also be considered as masters or slaves when performing data transfers (see Figure below).

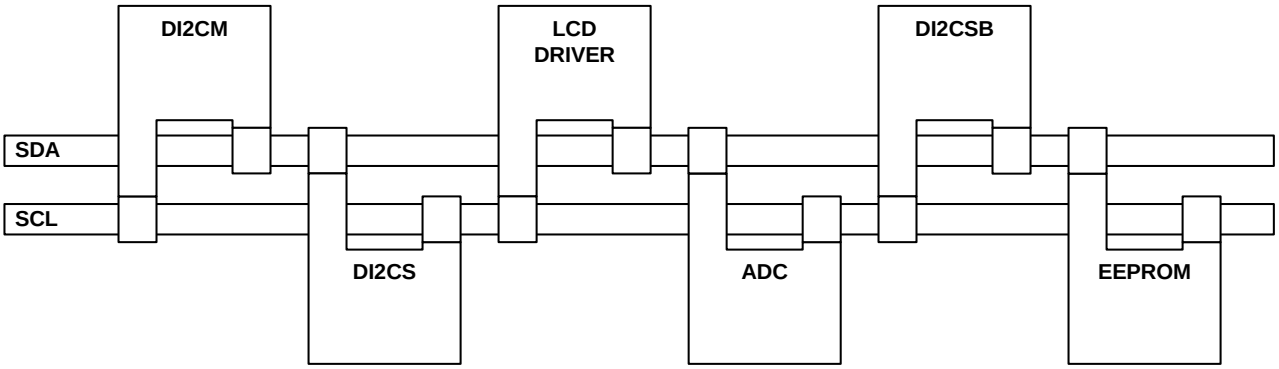


Figure-26. An example of I2C-bus application

A master is the device which initiates a data transfer on the bus and generates the SCL clock signals. The I2C-bus is a multi-master bus. This means that more than one device capable of controlling the bus can be connected to it. As masters are usually micro-controllers or microprocessors.

Definition Of I2c-Bus Terminology

Term	Description
Transmitter	The device which sends the data to the bus
Receiver	The device which receives the data from the bus
Master	The device which initiates a transfer, generates clock signals and terminates a transfer
Slave	The device addressed by a master
Multi-master	More than one mater can attempt to control the bus at the same time without corrupting the message
Arbitration	Procedure to ensure that, if more than one master simultaneously tries to control the bus, only one is allowed to do so and the message is not corrupted
Synchronization	Procedure to synchronize the clock signals of two or more devices

15.1.2 TRANSFERRING DATA

Both SDA and SCL are bi-directional lines, connected to a positive supply voltage via a pull-up resistor (see figure below). When the bus is free, both lines are HIGH. The output stages of devices connected to the bus must have an open drain or open-collector in order to perform the wired-AND function. Data on the I2C-bus can be transferred at a rate up to 100 kbit/s in the Standard mode, up to 400 kbit/s in the Fast mode.

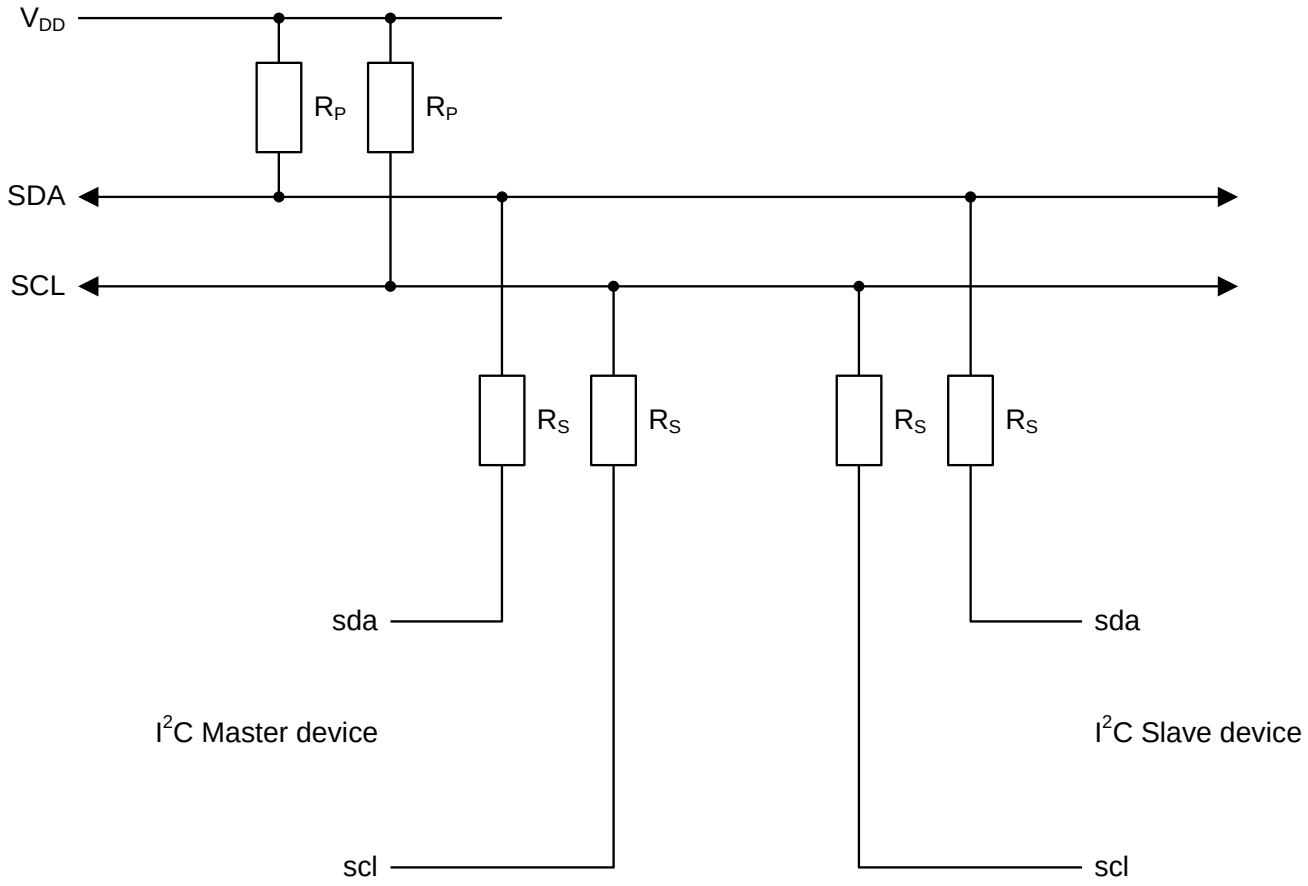


Figure-27. Connection Master And Slave Devices To I2c-Bus

15.1.3 BIT TRANSFER

Due to the variety of different technology devices (CMOS, NMOS, bipolar) which can be connected to the I2C-bus, the levels of the logical ‘0’ (LOW) and ‘1’ (HIGH) are not fixed and depend on the associated level of VDD. One clock pulse is generated for each data bit transferred.

15.1.4 DATA VALIDITY

The data on the SDA line must be stable during the HIGH period of the clock. The HIGH or LOW state of the data line can only change when the clock signal on the SCL line is LOW (see figure below).

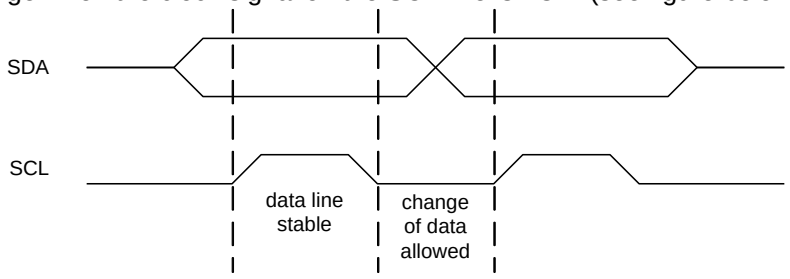


Figure-28. Data Validity During Bit Transfer On The I2C-Bus

15.1.5 START AND STOP CONDITIONS

Within the procedure of the I2C-bus, unique situations arise which are defined as START and STOP conditions (see figure below). A HIGH to LOW transition on the SDA line while SCL is HIGH is one such unique case. This situation indicates a START condition. A LOW to HIGH transition on the SDA line while SCL is HIGH defines a STOP condition. START and STOP conditions are always generated by the Master. The bus is considered to be busy after the START condition. The bus is considered to be free again a certain time after the STOP condition.

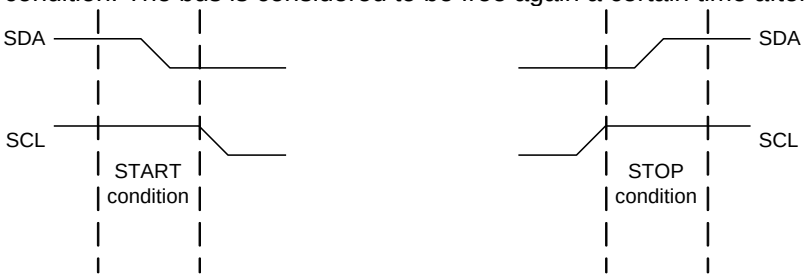


Figure-29. START And STOP Conditions

15.1.6 BYTE FORMAT

Every byte put on the SDA line must be 8-bits long. The number of bytes that can be transmitted per transfer is unrestricted. Each byte has to be followed by an acknowledge bit. Data is transferred with the most significant bit (MSB) first. If a receiver can't receive another complete byte of data until it has performed some other function, for example servicing an internal interrupt, it can hold the clock line SCL LOW to force the transmitter into a wait state. Data transfer than continues when the receiver is ready for another byte of data and releases clock line SCL.

15.1.7 ACKNOWLEDGE

Data transfer with acknowledge is obligatory. The acknowledge-related clock pulse is generated by the master. The transmitter releases the SDA line (HIGH) during the acknowledge clock pulse. The receiver must pull down the SDA line during the acknowledge clock pulse so that it remains stable LOW during the HIGH period of this clock pulse. Of course, set-up and hold times must also be taken into account.

When a slave-receiver doesn't acknowledge the slave address (for example, it's unable to receive because it's performing some real-time function), the data line must be left HIGH by the slave. The master can then generate a STOP condition to abort the transfer. If a master-receiver is involved in a transfer, it must signal the end of data to the slave-transmitter by not generating acknowledge on the last byte that was clocked out of the slave. The slave-transmitter must release the data line to allow the master to generate a STOP or repeated START condition.

15.1.8 CLOCK SYNCHRONIZATION

All masters generate their own clock on the SCL line to transfer messages on the I2C-bus. Data is only valid during the HIGH period of the clock. A defined clock is therefore needed for the bit-by-bit arbitration procedure to take place. Clock synchronization is performed using the wired-AND connection of I2C interfaces to the SCL line. A synchronized SCL clock is generated with its LOW period determined by the device with the longest clock LOW period, and its HIGH period determined by the one with the shortest clock HIGH period.

15.1.9 ARBITRATION

A master may start a transfer only if the bus is free. Two or more masters may generate a START condition within the minimum hold time of the START condition which results in a defined START condition to the bus. Arbitration takes place on the SDA line, while the SCL line is at the HIGH level, in such a way that the master which transmits a HIGH level, while another master is transmitting a LOW level will switch off its DATA output stage because the level on the bus doesn't correspond to its own level.

Arbitration can continue for many bits. Its first stage is comparison of the address bits. If the masters are each trying to address the same device, arbitration continues with comparison of the data-bits. Because address and data information on the I2C-bus is determined by the winning master, no information is lost during the arbitration process. A master that loses the arbitration can generate clock pulses until the end of the byte in which it loses the arbitration.

As an Hs-mode master has a unique 8-bit master code, it will always finish the arbitration during the first byte.

15.1.10 DATA FORMAT WITH 7-BIT ADDRESS

Data transfers follow the format shown in figure below. After the START condition, a slave address is sent. This address is 7 bits long followed by an eighth bit which is a data direction bit (R/S) - a 'zero' indicates a transmission (SEND), a 'one' indicates a request for data (RECEIVE). A data transfer is always terminated by a STOP condition generated by the master. However, if a master still wishes to communicate on the bus, it can generate a repeated START condition and address another slave without first generating a STOP condition. Various combinations of receive/send formats are then possible within such a transfer.

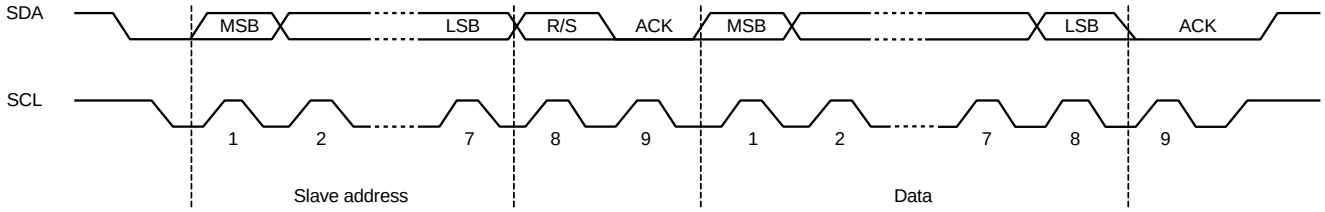


Figure-30. A Complete Data Transfer With 7-Bit Address

The first seven bits of the first byte make up the slave address. The eighth bit determines the direction of the message. A 'zero' in the R/S position of the first byte means that the master will write (send) information to a selected slave. A 'one' in this position means that the master will receive information from the slave.

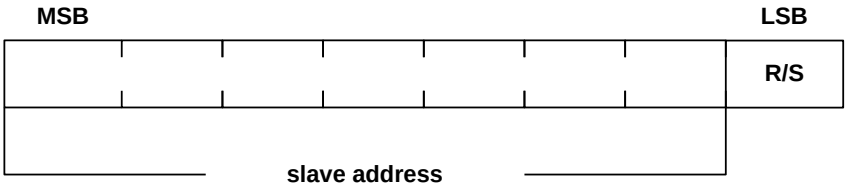


Figure-31. The First Byte After The START Condition

15.2 Internal Registers

There are six registers used to interface to the host: the Control, Status, Slave Address, Transmitted Data, Received Data and Timer Period Register.

DI2CM Register for Writing

Register	Address	Function
I2CMSA	0XF4	Slave address
I2CMCR	0XF5	Control
I2CMBUF	0XF6	Transmitter data register
I2CMTP	0XF7	Timer period

DI2CM Register for Reading

Register	Address	Function
I2CMSA	0XF4	Slave address
I2CMCR	0XF5	Status
I2CMBUF	0XF6	Receiver data register
I2CMTP	0XF7	Timer period

15.2.1 I2CMTP - TIMER PERIOD REGISTER

DI2CM Timer Period Register

I2CTP (r/w) : (SFR 0XF7) DI2CM Timer Period Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	CTP.7	CTP.6	CTP.5	CTP.4	CTP.3	CTP.2	CTP.1	CTP.0
Default	0	0	0	0	0	0	0	0

To generate wide range of SCL frequencies the core have build-in 8-bit timer.

SCL_PERIOD = 8 (1+TIMER_PRD) * CLK_PRD*

for example:

- CLK_PRD = 50ns (system clock freuqncy = 20 MHz);
- TIMER_PRD = 24; (The value contains in DI2CM Timer Period Register)
SCL_PERIOD = 8 (1+24)* 50ns = 8*25*50ns = 10000ns =10us*
SCL_FREQUENCY = 1/10us = 100kHz

15.2.2 I2CMCR - CONTROL AND STATUS REGISTERS

The Control Register consists of three bits: the RUN bit, the START bit, and the STOP bit. The START bit will cause the generation of the START, or REPEATED START protocol. The Stop bit determines if the cycle will stop at the end of the data cycle, or continue on to a burst. To generate a single read cycle, the Slave Address register is written with the desired address, the R/S bit is set to '1', and Control Register is written with ACK=0, STOP=1, START=1, RUN=1 (binary xxxx0111 x mean 0 or 1) to perform the operation and stop. When the operation is completed (or aborted due an error) the interrupt pin becomes active and the data may be read from Received Data Register. The ACK bit must be set normally to logic 1. This cause the I2C bus controller to send an acknowledge automatically after each byte.

The bit must be reset when the I2C bus controller is operated in master/receiver mode and requires no further data to be sent from slave transmitter.

DI2CM Control Register

I2CMCR (w) : (SFR 0XF5) DI2CM Control Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	-	-	-	HS	ACK	STOP	START	RUN
Default	-	-	-	0	0	0	0	0

Control Bits Combinations Permitted In IDLE State

HS	R/S	ACK	STOP	START	RUN	OPERATION
0	0	-	0	1	1	START condition followed by SEND (Master remains in Transmitter mode)
0	0	-	1	1	1	START condition followed by SEND and STOP condition
0	1	0	0	1	1	START condition followed by RECEIVE operation with negative Acknowledge *(Master remains in Receiver mode)*
0	1	0	1	1	1	START condition followed by RECEIVE and STOP condition
0	1	1	0	1	1	START condition followed by RECEIVE *(Master remains in Receiver mode)*
0	1	1	1	1	1	forbidden sequence

--- *All other control bits combinations not mentioned in table above are NOP's.*

Control Bits Combinations Permitted In Master Transmitter Mode

HS	R/S	ACK	STOP	START	RUN	OPERATION
0	-	-	0	0	1	SEND operation *(Master remains in Transmitter mode)*
0	-	-	1	0	0	STOP condition
0	-	-	1	0	1	SEND followed by STOP condition
0	0	-	0	1	1	Repeated START condition followed by SEND *(Master remains in Transmitter mode)*
0	0	-	1	1	1	Repeated START condition followed by SEND and STOP condition
0	1	0	0	1	1	Repeated START condition followed by RECEIVE operation with ***negative Acknowledge*** *(Master remains in Receiver mode)*
0	1	0	1	1	1	Repeated START condition followed by SEND and STOP condition
0	1	1	0	1	1	Repeated START condition followed by RECEIVE *(Master remains in Receiver mode)*
0	1	1	1	1	1	**forbidden sequence**

--- *All other control bits combinations not mentioned in table above are NOP's.*

Control Bits Combinations Permitted In Master Receiver Mode

HS	R/S	ACK	STOP	START	RUN	OPERATION
0	-	0	0	0	1	RECEIVE operation with ***negative Acknowledge*** *(Master remains in Receiver mode)*
0	-	-	1	0	0	STOP condition**
0	-	0	1	0	1	RECEIVE followed by STOP condition
0	-	1	0	0	1	RECEIVE operation *(Master remains in Receiver mode)*
0	-	1	1	0	1	**forbidden sequence**
0	1	0	0	1	1	Repeated START condition followed by RECEIVE operation with ***negative Acknowledge*** *(Master remains in Receiver mode)*
0	1	0	1	1	1	Repeated START condition followed by RECEIVE and STOP condition
0	1	1	0	1	1	Repeated START condition followed by RECEIVE *(Master remains in Receiver mode)*
0	0	-	0	1	1	Repeated START condition followed by SEND *(Master remains in Receiver mode)*
0	0	-	1	1	1	Repeated START condition followed by SEND and STOP condition

--- *All other control bits combinations not mentioned in table above are NOP's. in Master Receiver mode STOP condition should be generated only after Data negative Acknowledge executed by Master or Address negative Acknowledge executed by Slave.*

The Status Register consists of six bits: the BUSY bit, the ERROR bit, the ADDR_ACK bit, the DATA_ACK bit, the ARB_LOST bit, and the IDLE bit.

DI2CM Status Register

I2CMCR (r) : (SFR 0XF5) DI2CM Status Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	-	BUS_BUSY	IDLE	ARB_LOST	DATA_ACK	ADDR_ACK	ERROR	BUSY
Default	-	0	0	0	0	0	0	0

BUS_BUSY – this bit indicates that the Bus is Busy, and access is not possible. This bit is set/reset by START and STOP conditions;
IDLE - this bit indicates that I2C Bus controller is in the IDLE state:
ARB_LOST - this bit indicates that due the last operation I2C Bus controller lost the arbitration;
DATA_ACK - this bit indicates that due the last operation transmitted data wasn't acknowledged;
ADDR_ACK - this bit indicates that due the last operation slave address wasn't acknowledged;
ERROR - this bit indicates that due the last operation an error occurred: slave address wasn't acknowledged, transmitted data wasn't acknowledged, or I2C Bus controller lost the arbitration;
BUSY - this bit indicates that I2C Bus controller receiving, or transmitting data on the bus, and other bits of Status register are no valid;

15.2.3 I2CMSA - SLAVE ADDRESS REGISTER

The Slave Address Register consists of eight bits: seven address bits (A6-A0), and Receive/not Send bit R/S. The R/S bit determines if the next operation will be a Receive (high), or Send (low).

DI2CM Slave Address Register

I2CMSA (r/w) : (SFR 0XF4) DI2CM Slave Address Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	A.6	A.5	A.4	A.3	A.2	A.1	A.0	R/S
Default	0	0	0	0	0	0	0	0

DI2CMBUF - RECEIVER AND TRANSMITTER REGISTERS

The Transmitter Data Register consists of eight data bits which will be sent on the bus due the next Send, or Burst Send operation. The first send bit is the D.7 (MSB).

DI2CM Transmitter Data Register

I2CMBUF (w) : (SFR 0XF6) DI2CM Transmitter Data Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	D.7	D.6	D.5	D.4	D.3	D.2	D.1	D.0
Default	0	0	0	0	0	0	0	0

The Receiver Data Register consists of eight data bits which have been received on the bus due the last Receive, or Burst Receive operation.

DI2CM Receiver Data Register

I2CMBUF (r) : (SFR 0XF6) DI2CM Receiver Data Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	D.7	D.6	D.5	D.4	D.3	D.2	D.1	D.0
Default	0	0	0	0	0	0	0	0

15.3 Available Command Sequences

The command flowcharts presented in this chapter are concerned to NORMAL and FAST transmission speeds. In HIGH speed mode a special MASTER CODE is required before processing into particular operation. It is described in Operations in High-speed mode subchapter.

15.3.1 SINGLE SEND

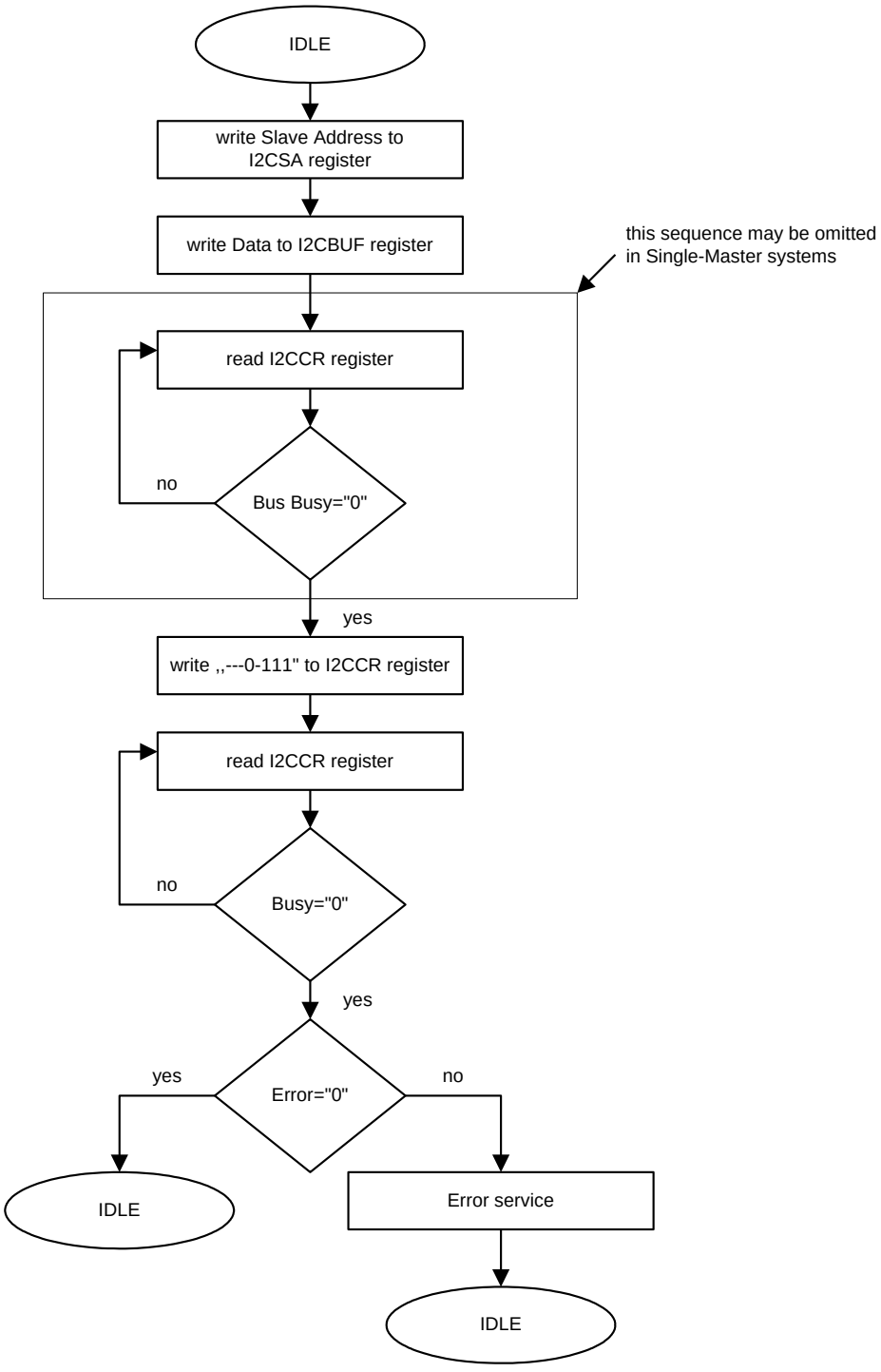


Figure-32. Single SEND Flowchart

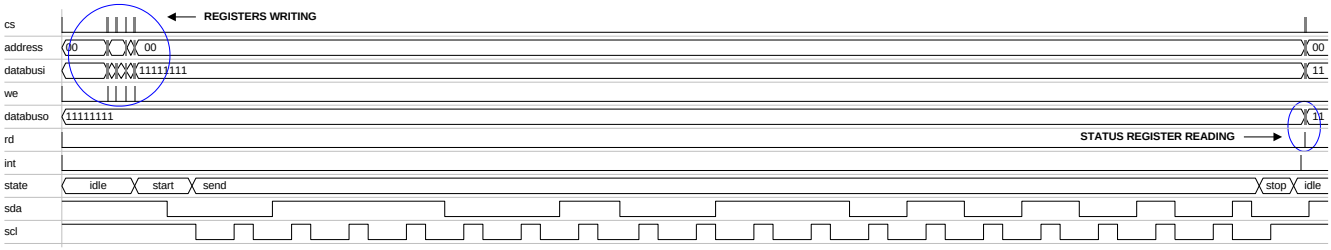


Figure-33. Single SEND Waveform

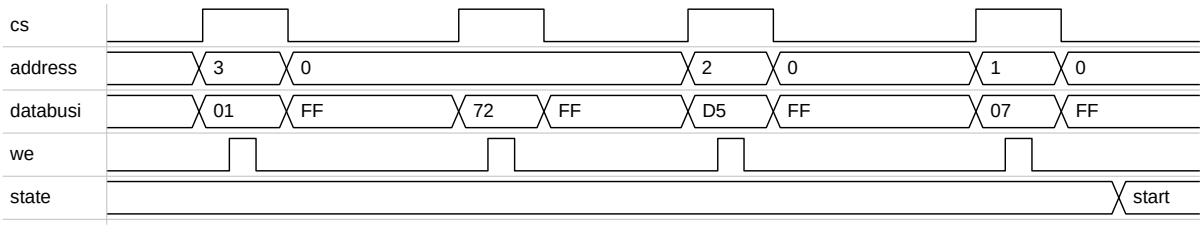


Figure-34. Single SEND Waveform - REGISTERS WRITING

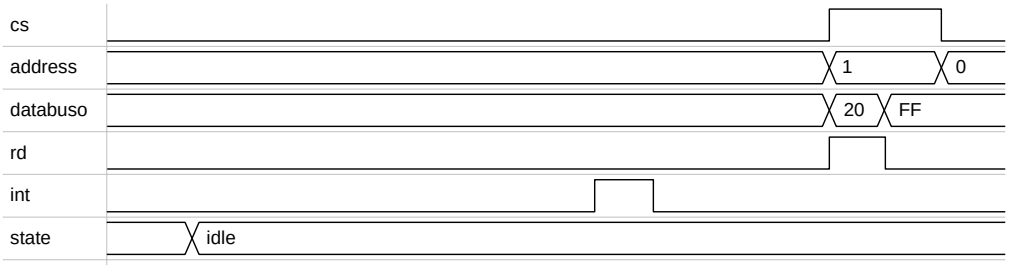


Figure-35. Single SEND Waveform - STATUS REGISTER READING

15.3.2 SINGLE RECEIVE

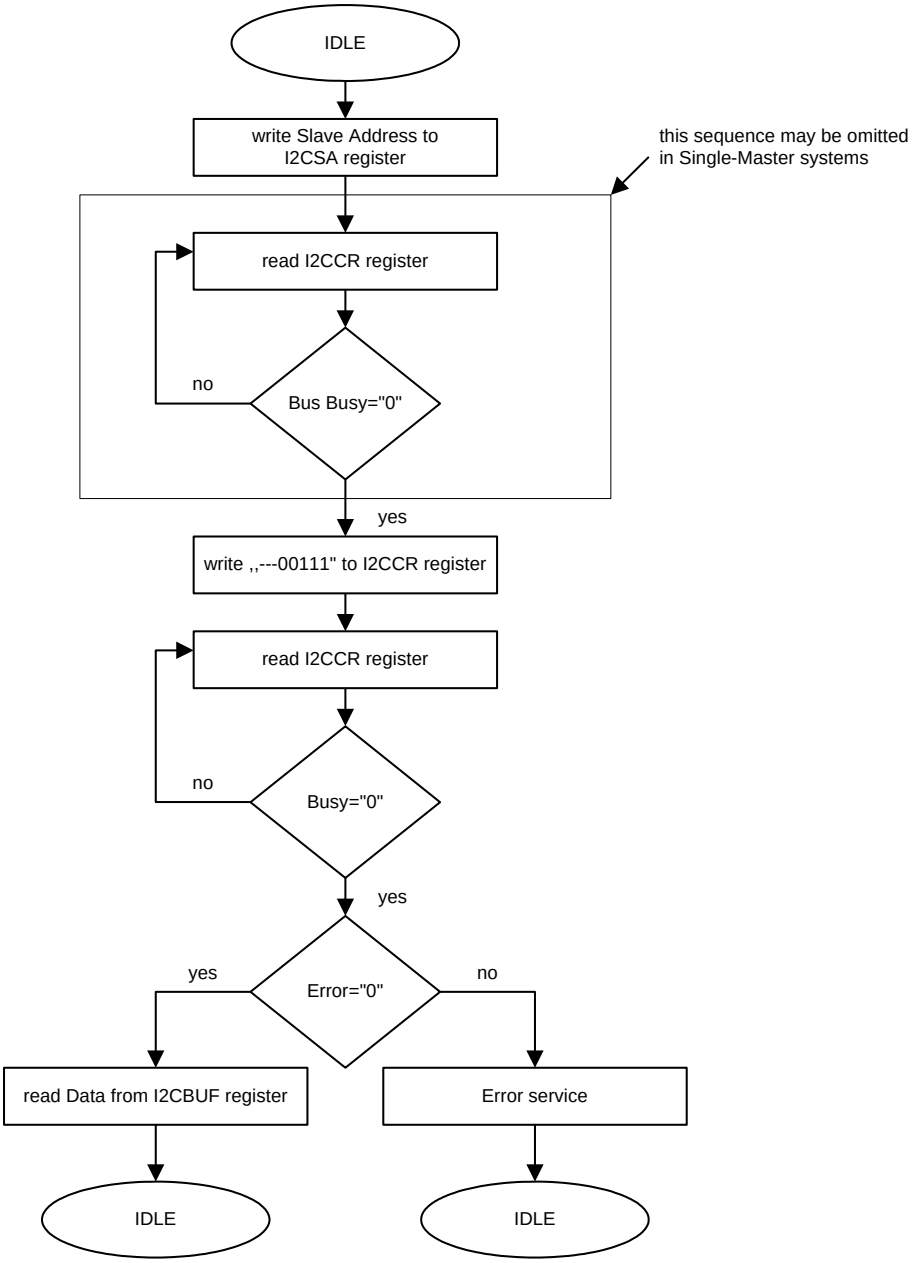


Figure-36. Single RECEIVER Flowchart

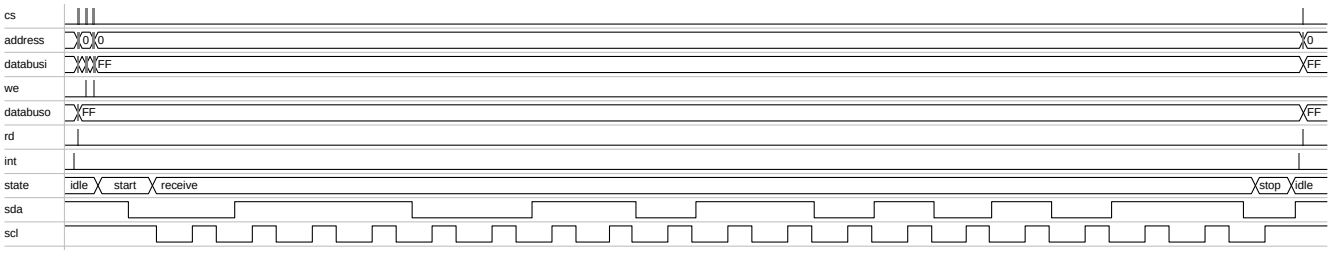


Figure-37. Single RECEIVE Waveform

15.3.3 BURST SEND

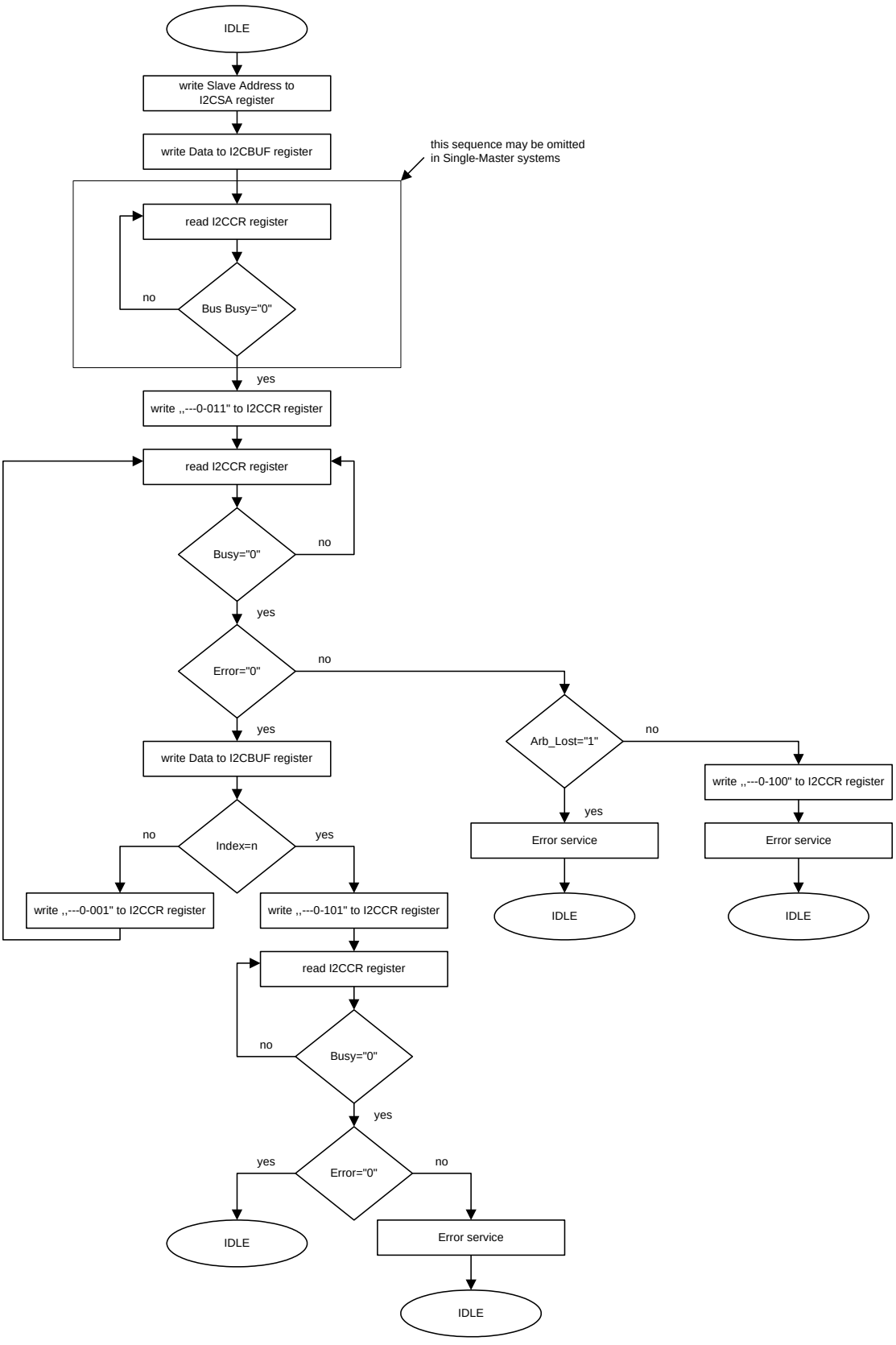


Figure-38. Sending n Bytes Flowchart

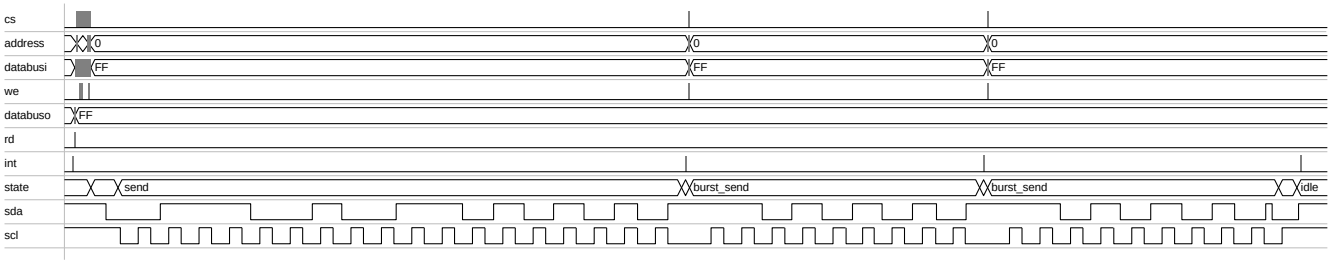


Figure-39. Sending n=3 Bytes Waveform

15.3.4 BURST RECEIVE

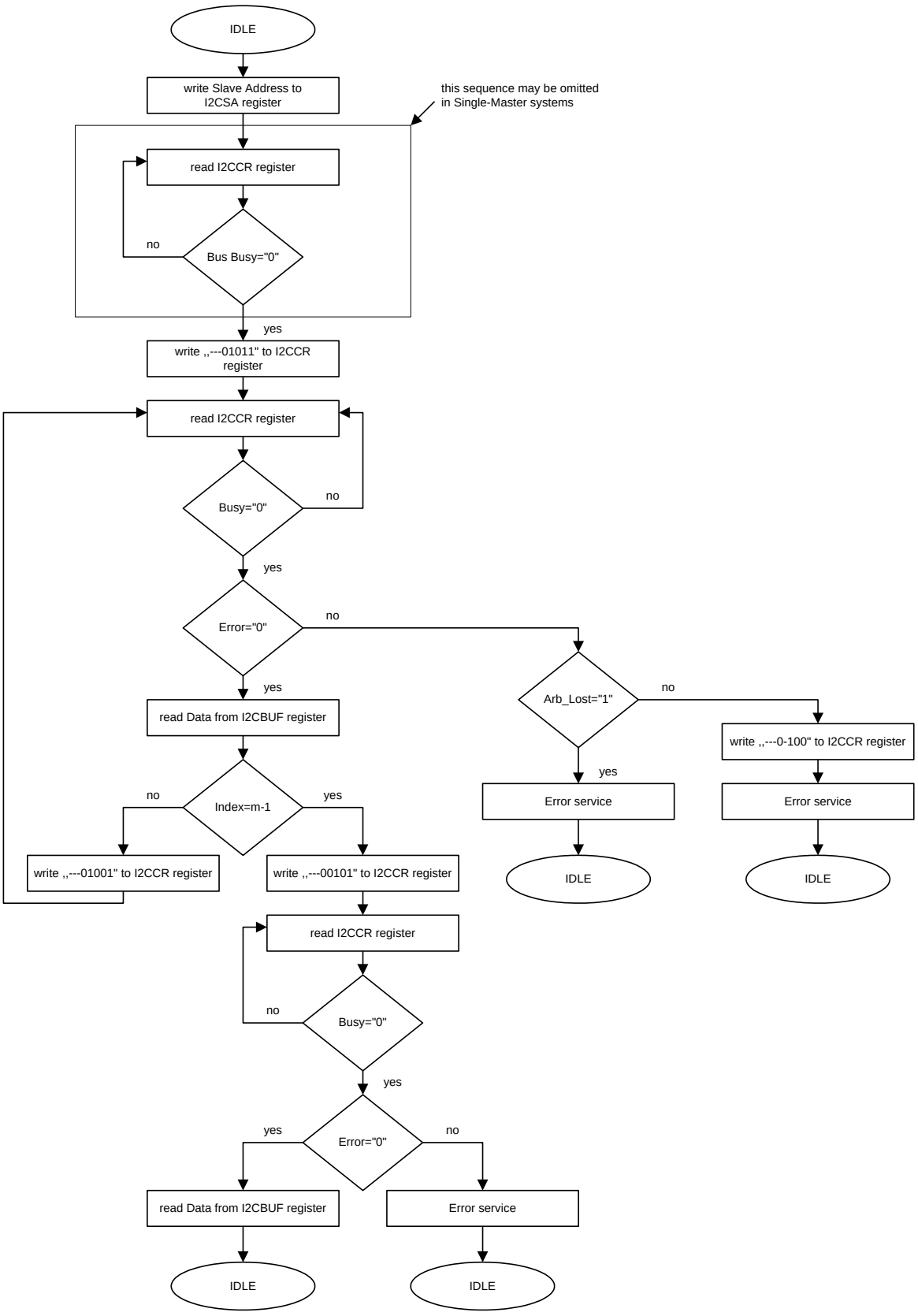


Figure-40. Receiving m Bytes Flowchart

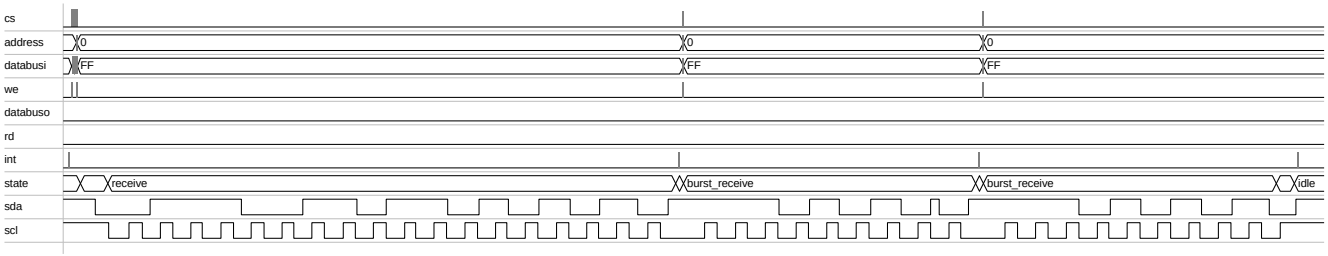


Figure-41. Receiving m=3 Bytes Waveform

15.3.5 BURST RECEIVE AFTER BURST SEND

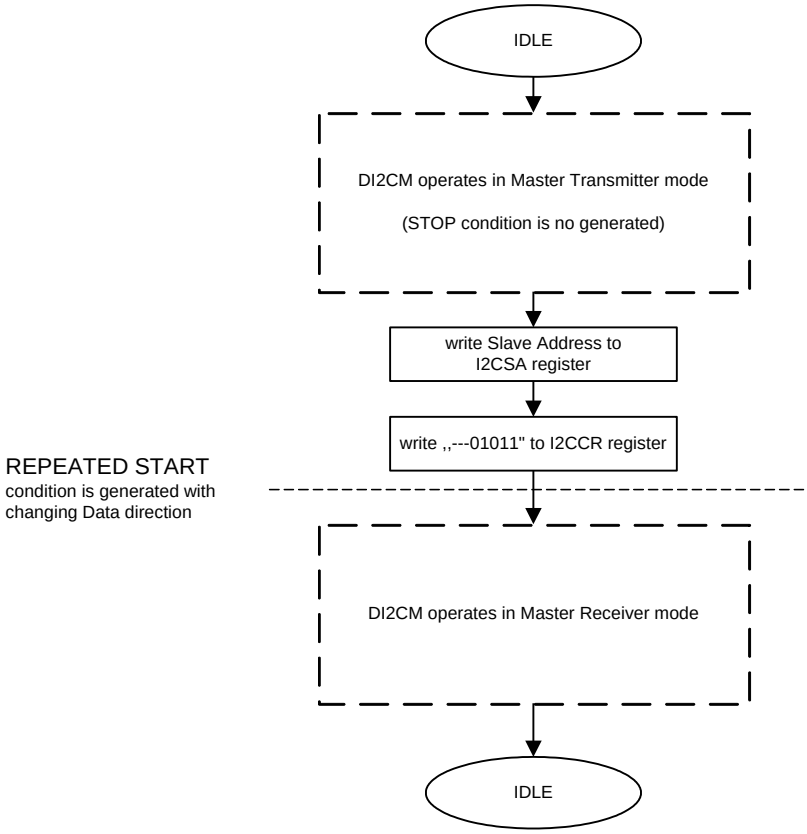


Figure-42. Sending n Bytes Followed By Repeated Start And Receiving m Bytes Flowchart

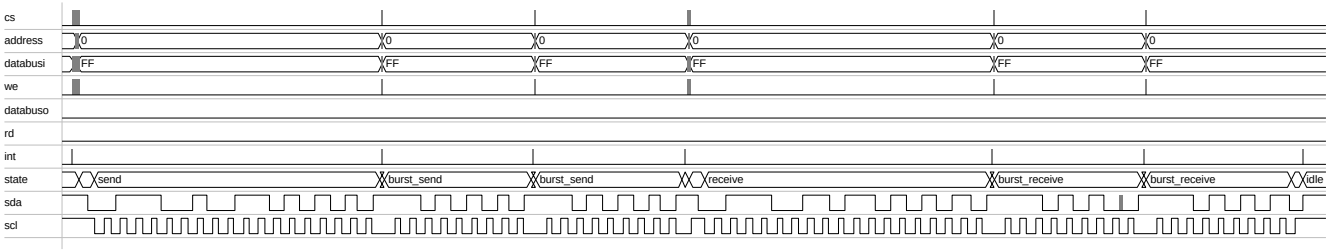


Figure-43. Sending 3 Bytes Followed By Repeated Start And Receiving 3 Bytes Waveform

15.3.6 BURST SEND AFTER BURST RECEIVE

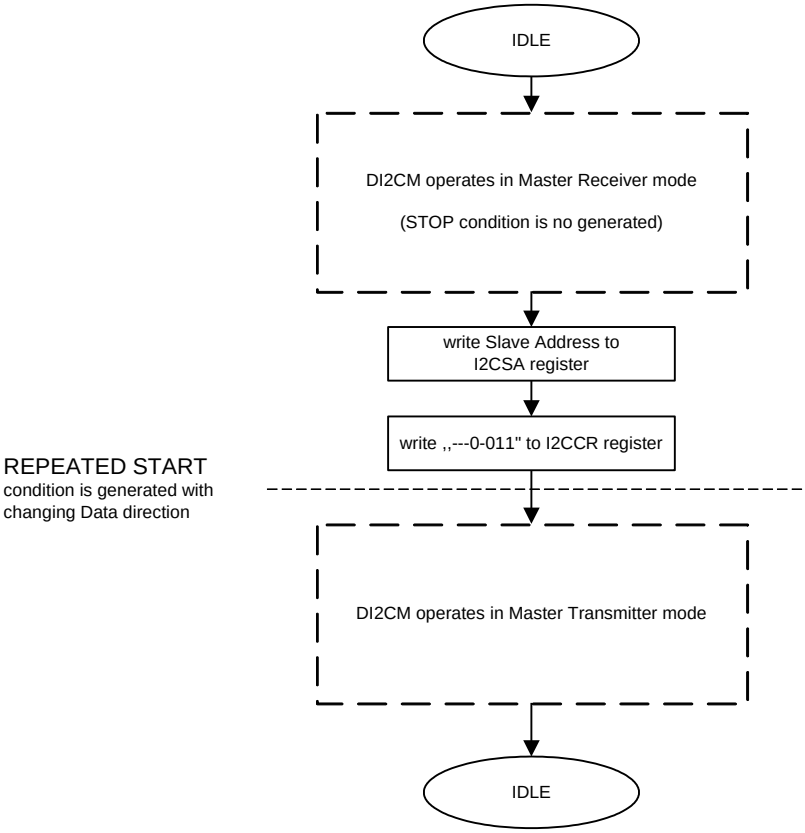


Figure-44. Receiving m Bytes Followed By Repeated Start And Sending n Bytes Flowchart

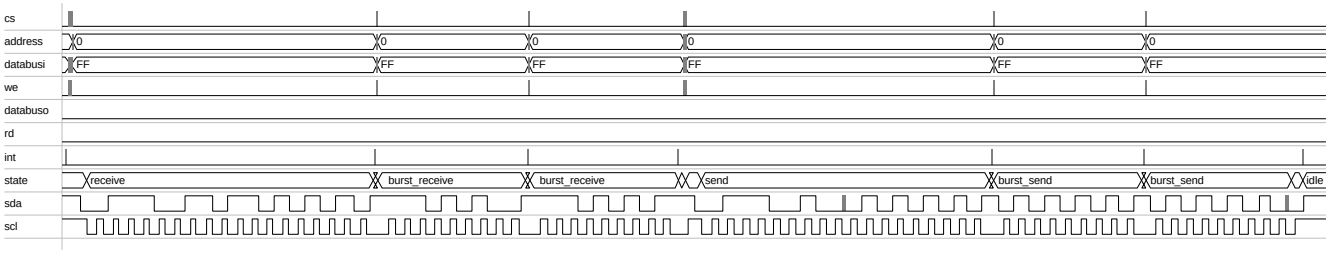


Figure-45. Receiving 3 Bytes Followed By Repeated Start And Sending 3 Bytes Waveform

16. SLAVE IIC

16.1 SlaveB 0 IIC And Slave B 1 IIC

The SlaveB 0 IIC block is connected to SDA0 and SCL0 (The SlaveB 1 IIC block is connected to SDA1 and SCL1) pins only. This block cans receive/transmit data using IIC protocols. S/W may write the SLVBADR register to determine the slave addresses.

In receive mode, the block first detects IIC slave address matching the condition then issues a SlvBMI0(SlvBMI1) interrupt. The data from SDA0(SDA1) is shifted into shift register then written to RCBBUF0(RCBBUF1) register when a data byte is received. The first byte loaded is word address (slave address is dropped). This block also generates a RCBI0(RCBI1) (receives buffer full interrupt) every time when the RCBBUF0(RCBBUF1) is loaded. If S/W is not able to read out the RCBBUF0(RCBBUF1) in time, the next byte in shift register is not written to RCBBUF0(RCBBUF1) and the slave block returns NACK to the master. This feature guarantees the data integrity of communication. The WadrB0(WadrB1) flag can tell S/W whether the data in RCBBUF0(RCBBUF1) is a word address or not.

In transmit mode, the block first detects IIC slave address matching the condition, then issues a SlvBMI0(SlvBMI1) interrupt. In the meantime, the data pre-stored in the TXBBUF0(TXBBUF1) is loaded into shift register, resulting in TXBBUF0(TXBBUF1) emptying and generates a TXBI0(TXBI1) (transmit buffer empty interrupt). S/W should write the TXBBUF0(TXBBUF1) a new byte for the next transfer before shift register empties. A failure of this process causes data corrupt. The TXBI0(TXBI1) occurs every time when shift register reads out the data from TXBBUF0(TXBBUF1).

The SlvBMI0(SlvBMI0) is cleared by writing "0" to corresponding bit in INTFLG0(INTFLG1) register. The RCBI0(RCBI1) is cleared by reading out RCBBUF0(RCBBUF1). The TXBI0(TXBI1) is cleared by writing TXBBUF0(TXBBUF1).

Slave B Register List

IICSTUS0	F01h (r)	WadrB0		SlvRWB0	SackIn0	SLVS0			
INTFLG0	F03h (r)	TXBI0	RCBI0	SlvBMI0	STOPI0	ReStaI0			
INTFLG0	F03h (w)			SlvBMI0	STOPI0	ReStaI0			
INTEN0	F04h (w)	ETXBI0	ERCBI0	ESlvBMI0	ESTOPI0	EreStaI0			
RCBBUF0	F08h (r)	Slave B 0 IIC receives buffer							
TXBBUF0	F08h (w)	Slave B 0 IIC transmits buffer							
SLVBADR0	F09h (w)	ENSlvB 0	Slave B 0 IIC address						
IICSTUS1	F91h(w)	WadrB1		SlvRWB1	Sackin1	SLVS1			
INTFLG1	F93h(r)	TXBI1	RCBI1	SlvBMI1	STOPI1	ReStal1			
INTFLG1	F93h(w)			SlvBMI1	STOPI1	ReStal1			
INTEN1	F94h(w)	ETXBI21	ERCBI1	ESlvBMI1	ESTOPI1	Erstal1			
RCBBUF1	F98h(r)	SlaveB 1 IIC receives buffer							
TXBBUF1	F98h(w)	SlaveB 1 IIC transimits buffer							
SLVBADR1	F99h(w)	ENSlvB1	SlaveB1 IIC address						

OPTION (r/w) : (XFR 0X0F56) Chip Option Configuration (default 0X00)

	Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Name	PWMF	DIV253	-	ENSCL1	ENSCL0	-	-	-
Default	0	0	-	0	0	-	-	-

ENSCL1 = 1 → Enable SlaveB 1 IIC block to hold SCL pin low while CS8959 is unable to catch-up with the external master's speed.

ENSCL0 = 1 → Enable SlaveB 0 IIC block to hold SCL pin low while CS8959 is unable to catch-up with the external master's speed.

16.2 SlaveB 0 IIC XFR

IICSTUS0 (r) : (XFR 0X0F01) IIC0 Status Registers (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	WadrB0	-	SlvRWB0	SackIn0	SLVS0	-	-	-
Default	0	-	0	0	0	-	-	-

WadrB0 =1 → The data in RCBBUF0 is word address.
SlvRWB0 =1 → Current transfer is slave transmit
=0 → Current transfer is slave receive
SackIn0 =1 → The external IIC host respond NACK.
SLVS0 =1 → The slave block has detected a START, cleared when STOP detected.

INTFLG0 (r/w) : (XFR 0X0F03) IIC0 Interrupt Flag Registers (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	TXBI0	RCBI0	SlvBMI0	STOPI0	ReStaI0	-	-	-
Default	0	0	0	0	0	-	-	-

An interrupt event will set its individual flag. While the corresponding interrupt enable bit is set, the AFI source will be driven by a zero level. Software MUST clear this register while serving the interrupt routine.

INTFLG0 (w) : Interrupt flag.
SlvBMI0 =1 → No action.
=0 → Clears SlvBMI0 flag.
STOPI0 =1 → No action.
=0 → Clears STOPI0 flag.
ReStaI0 =1 → No action.
=0 → Clears ReStaI0 flag.

INTFLG0 (r) : Interrupt flag.
TXBI0 =1 → Indicates the TXBBUF0 need a new data byte, cleared by writing TXBBUF0.
RCBI0 =1 → Indicates the RCBBUF0 has received a new data byte, cleared by reading RCBBUF0.
SlvBMI0 =1 → Indicates the slave IIC address B match condition.
STOPI0 =1 → Indicates the slave IIC has detected a STOP condition for SCL0/SDA0 pins.
ReStaI0 =1 → Indicates the slave IIC has detected a repeat START condition for SCL0/SDA0 pins.

INTEN0 (w) : (XFR 0X0F04) IIC0 Interrupt Enable Registers (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	ETXBI0	ERCBI0	ESlvBMI0	ESTOPI0	EreStaI0	-	-	-
Default	0	0	0	0	0	-	-	-

ETXBI0 = 1 → Enables TXBBUF0 interrupt.
ERCBI0 = 1 → Enables RCBBUF0 interrupt.
ESlvBMI0 = 1 → Enables slave address B 0 match interrupt.
ESTOPI0 = 1 → Enables IIC bus STOP interrupt.
EreStaI0 = 1 → Enables IIC bus repeat START interrupt.

RCBBUF0 (r) : (XFR 0X0F08) IIC0 Receive Data Buffer (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	IIC0RX7	IIC0RX6	IIC0RX5	IIC0RX4	IIC0RX3	IIC0RX2	IIC0RX1	IIC0RX0
Default	0	0	0	0	0	0	0	0

Slave IIC block B 0 receives data buffer.

TXBBUF0 (w) : (XFR 0X0F08) IIC0 Transmit Data Buffer (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	IIC0TX7	IIC0TX6	IIC0TX5	IIC0TX4	IIC0TX3	IIC0TX2	IIC0TX1	IIC0TX0
Default	0	0	0	0	0	0	0	0

Slave IIC block B 0 transmits data buffer.

SLVBADR0 (w) : (XFR 0X0F09) IIC0 Enable and Address Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	ENSlvB 0	IIC0A6	IIC0A5	IIC0A4	IIC0A3	IIC0A2	IIC0A1	IIC0A0
Default	0	0	0	0	0	0	0	0

EnslvB0 =1 → Enables slave IIC block B 0.
=0 → Disables slave IIC block B 0.
IIC0A6-0 → Slave IIC address B 0 to which the slave block should respond.

16.3 Slave B 1 IIC XFR

IICSTUS1 (r) : (XFR 0X0F91) IIC1 Status Registers (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	WadrB1	-	SlvRWB1	SackIn1	SLVS1	-	-	-
Default	0	-	0	0	0	-	-	-

WadrB1 =1 → The data in RCBBUF0 is word address.
SlvRWB1 =1 → Current transfer is slave transmit
=0 → Current transfer is slave receive
SackIn1 =1 → The external IIC host respond NACK.
SLVS1 =1 → The slave block has detected a START, cleared when STOP detected.

INTFLG1 (r/w) : (XFR 0X0F93) IIC1 Interrupt Flag Registers (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	TXBI1	RCBI1	SlvBMI1	STOPI1	ReStaI1	-	-	-
Default	0	0	0	0	0	-	-	-

An interrupt event will set its individual flag. While the corresponding interrupt enable bit is set, the AFI source will be driven by a zero level. Software MUST clear this register while serving the interrupt routine.

INTFLG1 (w) : Interrupt flag.

SlvBMI1	=1	→ No action.
	=0	→ Clears SlvBMI1 flag.
STOPI1	=1	→ No action.
	=0	→ Clears STOPI1 flag.
ReStaI1	=1	→ No action.
	=0	→ Clears ReStaI1 flag.

INTFLG1 (r) : Interrupt flag.

TXBI1	=1	→ Indicates the TXBBUF1 need a new data byte, cleared by writing TXBBUF1.
RCBI1	=1	→ Indicates the RCBBUF1 has received a new data byte, cleared by reading RCBBUF1.
SlvBMI1	=1	→ Indicates the slave IIC address B match condition.
STOPI1	=1	→ Indicates the slave IIC has detected a STOP condition for SCL1/SDA1 pins.
ReStaI1	=1	→ Indicates the slave IIC has detected a repeat START condition for SCL1/SDA1 pins.

INTEN1 (w) : (XFR 0X0F94) IIC1 Interrupt Enable Registers (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	ETXBI1	ERCBI1	ESlvBMI1	ESTOPI1	EreStaI1	-	-	-
Default	0	0	0	0	0	-	-	-

ETXBI1	= 1	→ Enables TXBBUF1 interrupt.
ERCBI1	= 1	→ Enables RCBBUF1 interrupt.
ESlvBMI1	= 1	→ Enables slave address B 1 match interrupt.
ESTOPI1	= 1	→ Enables IIC bus STOP interrupt.
EreStaI1	= 1	→ Enables IIC bus repeat START interrupt.

RCBBUF1 (r) : (XFR 0X0F98) IIC1 Receive Data Buffer (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	IIC1RX7	IIC1RX6	IIC1RX5	IIC1RX4	IIC1RX3	IIC1RX2	IIC1RX1	IIC1RX0
Default	0	0	0	0	0	0	0	0

Slave IIC block B 1 receives data buffer.

TXBBUF1 (w) : (XFR 0X0F98) IIC1 Transmit Data Buffer (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	IIC1TX7	IIC1TX6	IIC1TX5	IIC1TX4	IIC1TX3	IIC1TX2	IIC1TX1	IIC1TX0
Default	0	0	0	0	0	0	0	0

Slave IIC block B 1 transmits data buffer.

SLVBADR1 (w) : (XFR 0X0F99) IIC1 Enable and Address Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	ENSlvB 1	IIC1A6	IIC1A5	IIC1A4	IIC1A3	IIC1A2	IIC1A1	IIC1A0
Default	0	0	0	0	0	0	0	0

EnslvB1	=1	→ Enables slave IIC block B 1.
	=0	→ Disables slave IIC block B 1.
IIC1A6-0		→ Slave IIC address B 1 to which the slave block should respond.

17. SPI

The Serial Peripheral Interface (SPI) is an enhanced synchronous serial hardware which is compatible with Motorola's SPI specifications. There are extra 4 entries read FIFO and 4 entries write FIFO available inter the on-chip SPI circuit. CS8959 SPI control pins shared with P7.2, P7.3, P7.4, and P7.5. Its registers defined are as following.

PADMOD0 (r/w): (XFR 0X0F50) Pad mode control registers 0. (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SPI_MOD	-	-	-	-	-	-	DA8E
Default	0	-	-	-	-	-	-	0

SPI_MOD = 1 → Pin "P7.2/ET2T/SCS" is SPI_SCS; pin "P7.3/ET2ET/MOSI" is SPI_MOSI.
→ Pin "P7.4/ET3T/MISO" is SPI_MISO; pin "P7.5/ET3ET/MSCK" is SPI_MSCK.
= 0 → Pin "P7.2/ET2T/SCS" is P7.2; pin "P7.3/ET2ET/MOSI" is P7.3.
→ Pin "P7.4/ET3T/MISO" is P7.4; pin "P7.5/ET3ET/MSCK" is P7.5.

DA8E = 1 → Pin "P6.4/PWM8" is PWM8.
= 0 → Pin "P6.4/PWM8" is P6.4.

SPICR (r/w): (SFR 0XEC) SPI Configuration Register. (default 0X20)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SPIE	SPIEN	MSTR	CPOL	CPHA	SCKE	-	-
Default	0	0	1	0	0	0	-	-

SPIE → SPI Interrupt Enable
= 1 → Enable SPI Interrupt.
= 0 → Disable SPI Interrupt.

SPIEN → SPI Enable
= 1 → Enable SPI function and interface
= 0 → Disable SPI function and interface

MSTR → SPI operation mode select.
= 1 → SPI operates in Master Mode.
= 0 → SPI operates in Slave Mode.

CPOL → SPI Clock Polarity Control Bit
= 1 → Allow SCK at 1 during SPI is stopped.
= 0 → Allow SCK at 0 during SPI is stopped.

CPHA → SPI Clock Phase Select Bit
= 1 → If CPOL = 1, Data transmitted on falling edge of SCK.
If CPOL = 0, Data transmitted on rising edge of SCK.
= 0 → If CPOL = 1, Data transmitted on rising edge of SCK.
If CPOL = 0, Data transmitted on falling edge of SCK.

SCKE → SPI Data Input Sample Phase
= 1 → Use rising edge of SCK to sample the input data
= 0 → Use falling edge of SCK to sample the input data

SPIMR (r/w): (SFR 0XED) SPI Mode Control Register. (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	ICNT[1:0]		FCLR	-	SPR[2:0]			DIR
Default	0		0	-	000			0

ICNT[1:0] → Interrupt threshold of transmit and receive byte count
= 00 → Interrupt is generated after 1 byte is sent or received

= 01 → Interrupt is generated after 2 bytes are sent or received
= 10 → Interrupt is generated after 3 bytes are sent or received
= 11 → Interrupt is generated after 4 bytes are sent or received

FCLR → FIFO Clear Bit
= 1 → Clear transmit and receive FIFO
= 0 → No action

SPR[2:0] → SPI Clock (SCK) Rate Setting
= 000 → SCK = SYSCLK / 6
= 001 → SCK = SYSCLK / 8
= 010 → SCK = SYSCLK / 16
= 011 → SCK = SYSCLK / 32
= 100 → SCK = SYSCLK / 64
= 101 → SCK = SYSCLK / 128
= 110 → SCK = SYSCLK / 256
= 111 → SCK = SYSCLK / 512

DIR → SPI Transfer Direction
= 1 → Transfer data in MSB first format.
= 0 → Transfer data in LSB first format.

SPIST (r/w): (SFR 0XEE) SPI Status Register. (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SSPIF	POVR	TOVR	TUDR	RFULL	REMPT	TFULL	REMPT
Default	0	0	0	0	0	0	0	0

SSPIF → SPI Interrupt Flag. Set by hardware to indicate that the data transfer has been completed. Clear by written 0 to this bit or disable SPI.

POVR → Receive FIFO overrun error flag. Receiving another data in receive FIFO full state, ROVR will be set and generate an interrupt. Clear by written 0 to this bit or disable SPI.

TOVR → Transfers FIFO overrun error flag. Writing another data in transfer FIFO full state, TOVR will be set and generate an interrupt. Clear by written 0 to this bit or disable SPI.

TUDR → Transmit under-run error flag. Transmitting another data in transfer FIFO empty state, TUDR will be set and generate an interrupt. Clear by written 0 to this bit or disable SPI.

RFULL → Receive FIFO Full Status
= 1 → Receive FIFO is full. Read only.
= 0 → Receive FIFO is not full. Read only.

REMPT → Receive FIFO Empty Status
= 1 → Receive FIFO is empty. Read only.
= 0 → Receive FIFO is not empty. Read only.

TFULL → Transfer FIFO Full Status
= 1 → Transfer FIFO is full. Read only.
= 0 → Transfer FIFO is not full. Read only.

REMPT → Transfer FIFO Empty Status
= 1 → Transfer FIFO is empty. Read only.
= 0 → Transfer FIFO is not empty. Read only.

SPIDATA (r/w): (SFR 0XEF) SPI Data Register. (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SPI Data Register							
Default	0X00							

SPIDATA → When writing, SPIDATA is for SPI transmit data.
→ When reading, SPIDATA is for SPI receive data.

17.1 SPI Timing Illustration

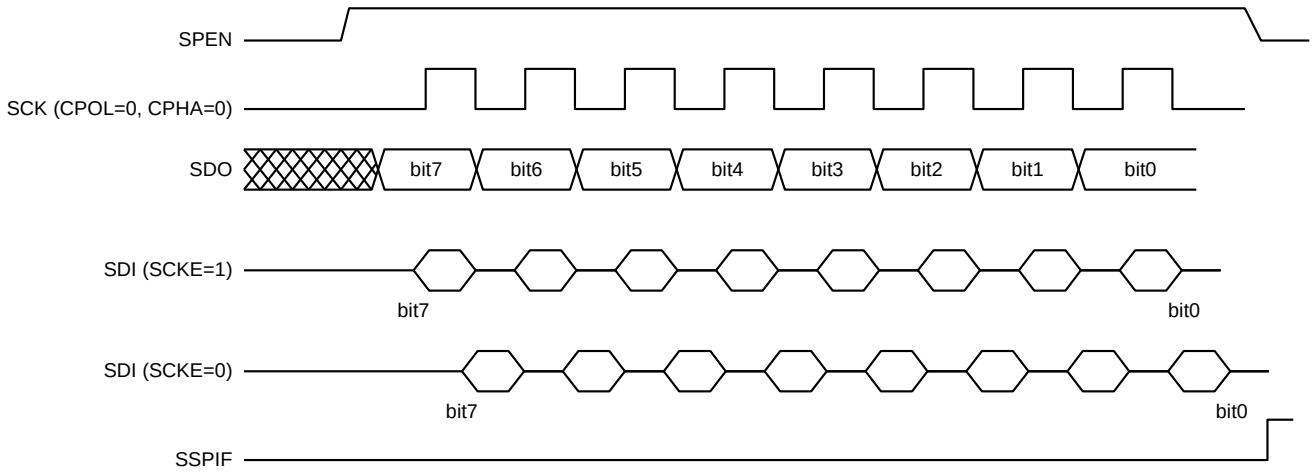


Figure-46. SPI Mode Timing, Master Mode

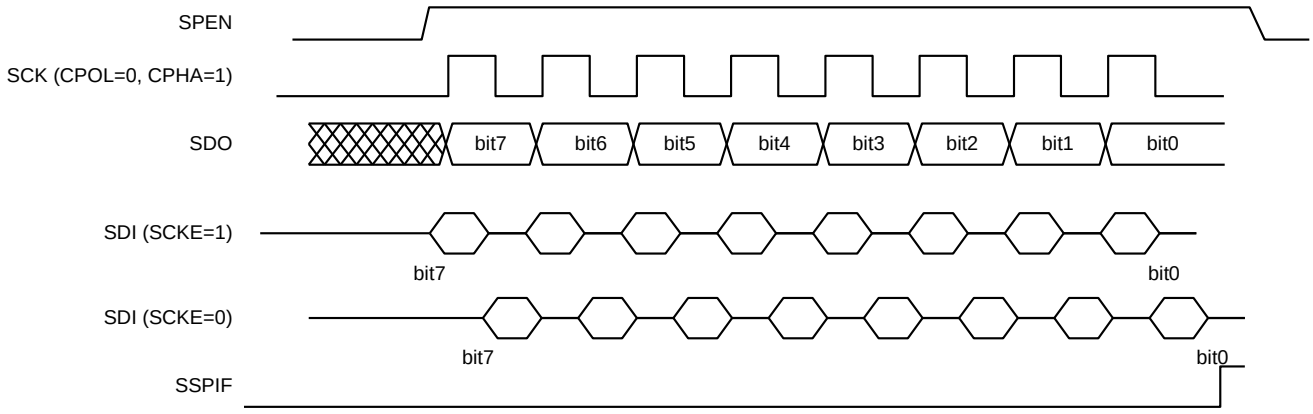


Figure-47. SPI Mode Timing, Master Mode

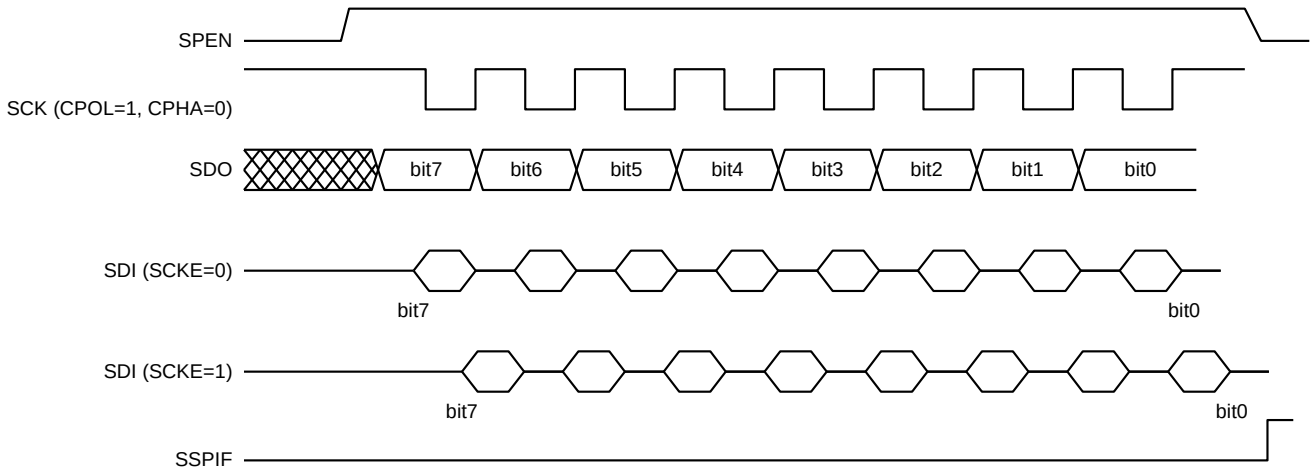


Figure-48. SPI Mode Timing, Master Mode

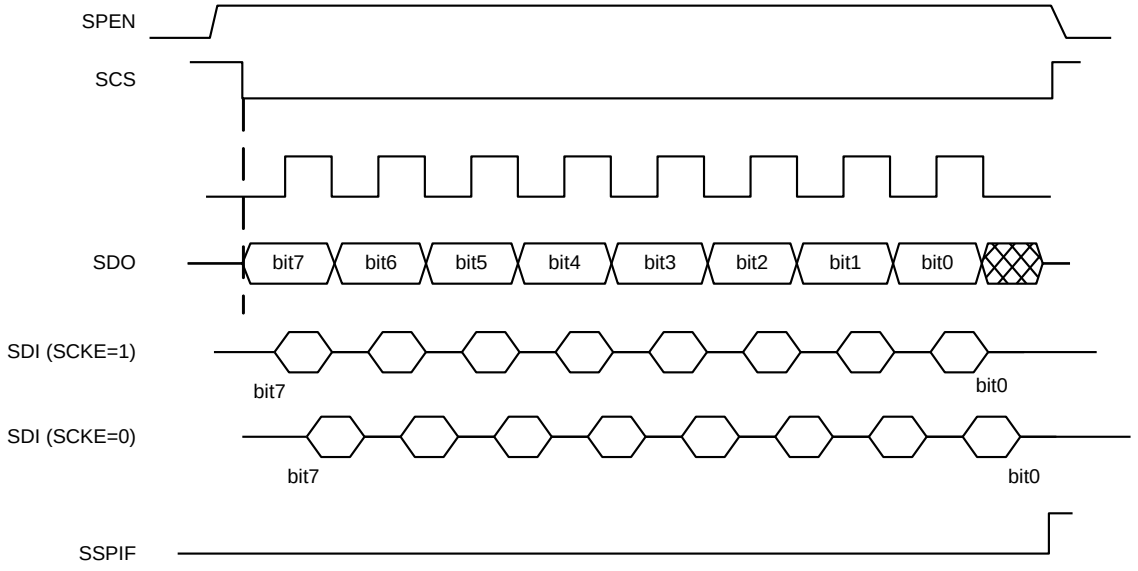


Figure-49. SPI Mode Timing, Slave Mode

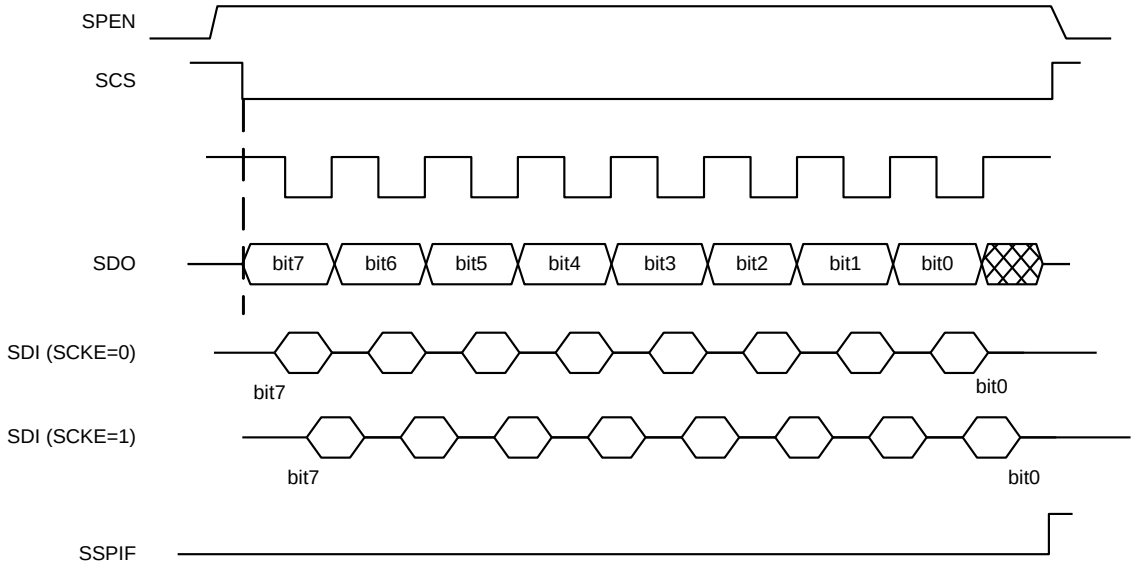


Figure-50. SPI Mode Timing, Slave Mode

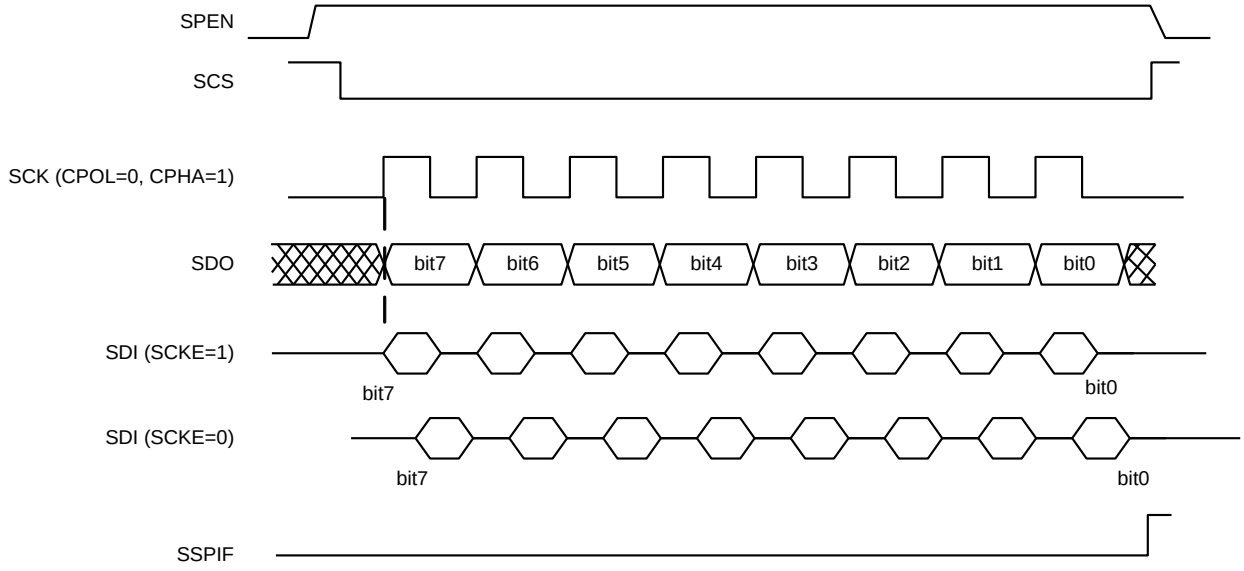


Figure-51. SPI Mode Timing, Slave Mode

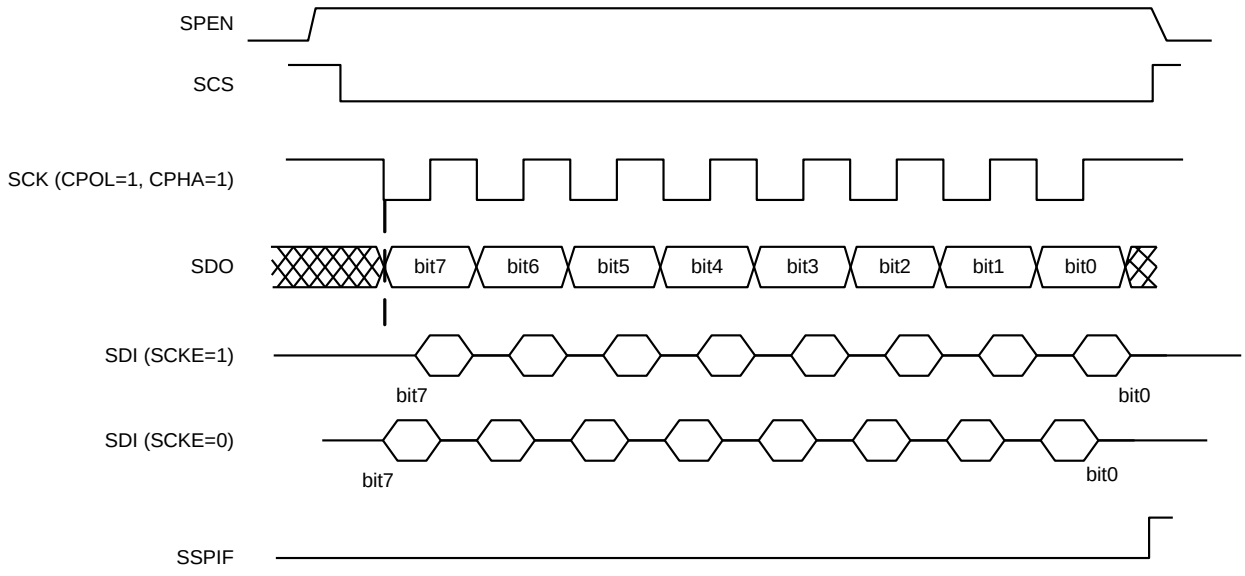


Figure-52. SPI Mode Timing, Slave Mode

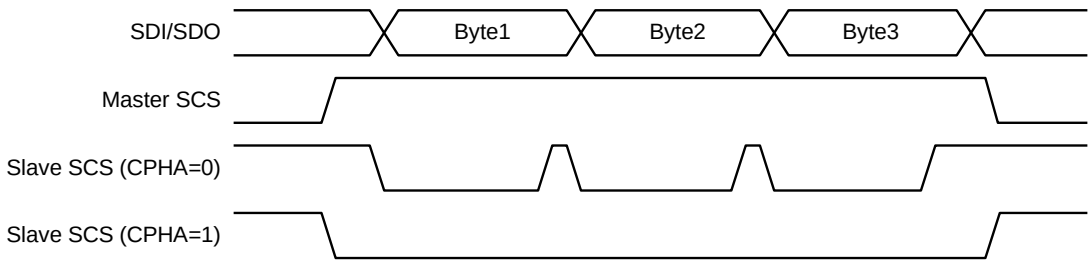


Figure-53. CPHA/SCS Timing

18. ISP

The Flash memory can be programmed by a specific writer (the user may need to import a .hex file) in serial mode via JTAG or UART while the system is working. The CRC check is feasible during ISP for error detection.

After Power on / Reset, the CS8959 is running the original program code; when ISP is enabling, the flash interface is switched to ISP host, and meanwhile the 8051 stops fetching new program data. To prevent unexpected MCU behavior, it is suggested S/W keeps the MCU in an idle status when entering ISP mode; moreover, the Watchdog Timer Reset function must be disabled.

ISP mode can be activated by software or hardware, described as the following:

1), S/W ISP – write ISPEN register to enter into ISP mode. (91h:JTAG ISP / 92h:UART ISP)
2), H/W ISP – a serial password (the same for JTAG ISP and UART ISP) is defined to avoid mistaken operation. CS8959 switches to ISP mode only when the valid “ISP Slave Address and Compared Data” are received from the JTAG or UART port. The default values* for the sequence are “0X94 0XAC 0XCA 0X53” in hexadecimal form.

ISP Registers (Write-Only)

Reg name	XFR Addr	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ISPSLV	0X0F0B(w)	ISP Slave Address						-	-
ISPEN	0X0F0C(w)	Write 0X91 / 0X92 to enable JTAG/UART ISP Mode							
ISPCMP1	0X0F0D(w)	ISP compared data 1							
ISPCMP2	0X0F0E(w)	ISP compared data 2							
ISPCMP3	0X0F0F(w)	ISP compared data 3							

ISPSLV: to define the ISP block address for identification. The valid ISP address consists of the six highest bits, and the two lowest bits are used for command indictor. The default value is 0X94.

ISPEN: S/W ISP Enable register. Write 0X91 to enable JTAG ISP mode / 0X92 to enable UART ISP mode. Default value = 0X00.

ISPCMP1: ISP compared data 1. The default value is 0XAC, and 0X00 is an invalid value.
ISPCMP2: ISP compared data 2. The default value is 0XCA, and 0X00 is an invalid value.
ISPCMP3: ISP compared data 3. The default value is 0X53, and 0X00 is an invalid value.

*It is recommended not to modify the sequence unless the specific writer is compatible with changeable “ISP address” or “compared data”.

18.1 JTAG ISP

The ISP function is implemented in one chain of the JTAG TAP (Test Access Port) controller. With specific writer software, the user can easily download his program to the MCU via the JTAG port. For more information about the JTAG TAP controller, please refer to the chapter of JTAG TAP CONTROLLER.

18.2 UART ISP

In CS8959, UART0 is allocated for UART ISP; if the UART ISP function is desired, P3.0(RX0) and P3.1(TX0) must be reserved for the UART host while activating UART ISP mode.

The baud rate of UART ISP is self-synchronized* to the host (4800, 9600 or 19200 bps is recommended), depending on the operating frequency. Refer to the table below.

Recommended baud rate under different operating frequencies

Baud Rate (bps) / Operating Frequency (MHz)	4800	9600	19200
3.5795	O	X	X
4.000	O	X	X
6.000	O	X	X
8.000	O	O	X
11.059	O	O	X
12.000	O	O	X
12.096	O	O	X
14.318	O	O	O
16.000	O	O	O
20.000	O	O	O
22.118	O	O	O
24.576	O	O	O
25.000	O	O	O

X : Not support in the UART writer
O: Baud rate is workable in the UART writer

*Since the baud rate of UART ISP, independent of the UART0 settings, will be synchronized to “the first data byte **0X55**” detected on P3.0; once the synchronization is established the value could not be changed until next “CPU Reset”, hence earlier data transferred over P3.0 could probably cause baud rate mismatch for UART ISP (fail to enter ISP mode).

Considering only UART data rate and hardware accessing time, the following table shows estimated time for ISP operations. Accessing time will be proportional to the system period, but UART baud rate is still the dominant factor. In real applications it will take longer due to information processing delay.

Estimated time for ISP operations (System Frequency = 20MHz)

UART Baud Rate	4800 bps	9600 bps	19200 bps
Read 1 byte	18.8 ms	9.4 ms	4.7 ms
Read 1 page (1024 bytes)	2168.4 ms	1093.4 ms	555.9 ms
Read whole flash (171k bytes)	368002.0 ms	185576.9 ms	94364.4 ms
Program 1 word (2 bytes)	20.9 ms	10.5 ms	5.3 ms
Program 1 page(1024 bytes)	2182.8 ms	1107.8 ms	570.3 ms
Program whole flash (171k bytes)	370453.0 ms	188028.3 ms	96815.8 ms
Read CRC code (1 time)	14.6 ms	7.3 ms	3.7 ms
Erase whole flash	357.0 ms	346.6 ms	341.4 ms

19. WATCHDOG TIMER

The Watchdog Timer is a user programmable clock counter that can serve as:
__ a time-base generator
__ an event timer,
__ or a system supervisor.

As can be seen in the figure below, the timer is driven by the main system clock that is supplied to a series of dividers. The divider output is selectable, and determines interval between timeouts. When the timeout is reached, an interrupt flag will be set, and if enabled, a reset will occur. The interrupt flag will cause an interrupt to occur if its individual enable bit is set and the global interrupt enable is set. The reset and interrupt are completely discrete functions that may be acknowledged or ignored, together or separately for various applications.

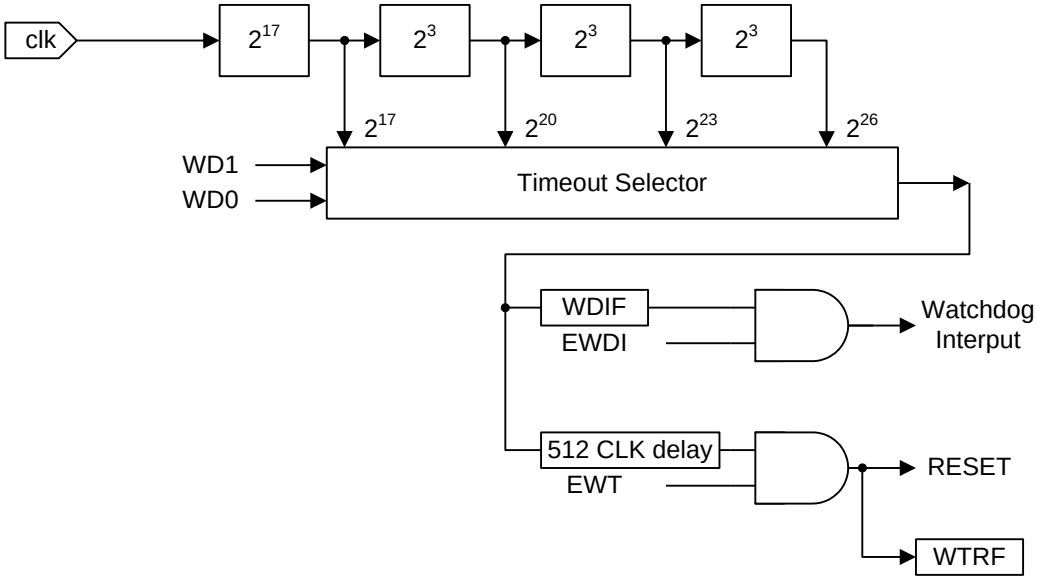


Figure-54. Watchdog Timer Structure

19.1 Watchdog Timer Reset

The Watchdog Timer Reset function works as follows. After initializing the correct timeout interval, software first restarts the Watchdog using RWT and then enables the reset mode by setting the Enable Watchdog Timer Reset (WDCON.1) bit. At any time prior to reaching its user selected terminal value, software can set the Reset Watchdog Timer (WDCON.0) bit. If RWT is set before the timeout is reached, the timer will start over. If the timeout is reached without RWT being set, the Watchdog will reset the CPU. Hardware will automatically clear RWT after software sets it. When the reset occurs, the Watchdog Timer Reset Flag (WDCON.2) will automatically be set to indicate the cause of the reset, however software must clear this bit manually.

19.2 Simple Timer

The Watchdog Timer is a free running timer. When used as a simple timer with both the reset (EWT=0) and interrupt functions disabled (EWDI=0), the timer will continue to set the Watchdog Interrupt flag each time the timer completes the selected timer interval as programmed by WD[1:0]. Restarting the timer using the RWT bit, allows software to use the timer in a polled timeout mode. The WDIF bit is cleared by software or any reset. The Watchdog Interrupt is also available for applications that don't need a true Watchdog Reset but simply a very long timer. The interrupt is enabled using the Enable Watchdog Timer Interrupt (EIE.4) bit. When the timeout occurs, the Watchdog Timer will set the WDIF bit (WDCON.3), and an interrupt will occur if the global interrupt enable (EA) is set. Note that WDIF is set 512 clocks before a potential Watchdog Reset. The Watchdog Interrupt Flag indicates the source of the interrupt, and must be cleared by software. Proper use of the Watchdog Interrupt with the Watchdog Reset allows interrupt software to survey the system for errant conditions.

19.3 System Monitor

When using the Watchdog Timer as a system monitor, the Watchdog Reset function should be used. If the Interrupt function was used, the purpose of the watchdog would be defeated. For example, assume the system is executing errant code prior to the Watchdog Interrupt. The interrupt would temporarily force the system back into control by vectoring the CPU to the interrupt service routine. Restarting the Watchdog and exiting by an RETI or RET, would return the processor to the lost position prior to the interrupt. By using the Watchdog Reset function, the processor is restarted from the beginning of the program, and therefore placed into a known state.

19.4 Watchdog Related Registers

The watchdog timer has several SFR bits that contribute to its operation. It can be enabled to function as either a reset source, interrupt source, software polled timer or any combination of the three. Both the reset and interrupt have status flags. The watchdog also has a bit that restarts the timer. A summary table showing the bit locations is below. A description follows.

Watchdog Related Bits Summary

Bit name	Register	Bit position	Description
EWDI	EIE	EIE.4	Enable Watchdog Timer Interrupt
WD[1:0]	CKCON	CKCON.7-6	Watchdog Interval
RWT	WDCON	WDCON.0	Reset Watchdog Timer
EWT		WDCON.1	Enable Watchdog Timer Reset
WTRF		WDCON.2	Watchdog Timer Reset flag
WDIF		WDCON.3	Watchdog Interrupt flag

19.4.1 WATCHDOG CONTROL

Watchdog control bits are described below. Please note that access (write) to this register has to be performed using Timed access registers procedure.

Watchdog Control Register

WDCON(w/r) : (SFR 0XD8) Watchdog Control Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	-	-	-	-	WDIF	WTRF	EWT	RWT
Default	-	-	-	-	0	0	0	0

WDIF - Watchdog Interrupt Flag. WDIF in conjunction with the Enable Watchdog Interrupt bit (EWDI:EXIE.4), and EWT, indicates if a watchdog timer event has occurred and what action should be taken. This bit must be cleared by software before exiting the interrupt service routine, or another interrupt is generated. Setting WDIF in software will generate a watchdog interrupt if enabled.

WTRF - Watchdog Timer Reset Flag. When set by hardware, indicates that a watchdog timer reset has occurred. Set by software don't generate a watchdog timer reset. It is cleared by RSTN pin, but otherwise must be cleared by software. The watchdog timer has no effect on this bit, If EWT bit is cleared.

EWT - Enable Watchdog Timer Reset. The reset of micro-controller by watchdog timer is controlled by this bit. This bit has no effect on the ability of the watchdog timer to generate a watchdog interrupt.
0 - watchdog timer timeout doesn't reset micro-controller
1 - watchdog timer timeout resets micro-controller

RWT - Reset Watchdog Timer. Setting RWT resets the watchdog timer count. Timed Access procedure must be used to set this bit before the watchdog timer expires, or a watchdog timer reset and/or interrupt will be generated if enabled.

- - Unimplemented bit. Read as 0 or 1.

Table below summarizes Watchdog Control bits and taken operation concerned to theirs values.

Watchdog Bits And Actions

EWT	EWDI	WDIF	Result
x	x	0	No watchdog event.
0	0	1	Watchdog time-out has expired. No interrupt has been generated.
0	1	1	Watchdog interrupt has occurred.
1	0	1	Watchdog time-out has expired. No interrupt has been generated. Watchdog timer reset will occur in 512 clock periods (CLK pin) if RWT is not strobed.
1	1	1	Watchdog interrupt has occurred. Watchdog timer reset will occur in 512 clock periods (CLK pin) if RWT is not set using Timed Access procedure.

19.4.2 CLOCK CONTROL

The Watchdog timeout selection is made using bits WD[1:0] as shown in the figure.

Clock Control Register ----Watchdog Bits

CKCON(w/r) : (SFR 0X8E) Clock Control Register (default 0X03)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	WD1	WD0	T2M	T1M	T0M	MD2	MD1	MD0
Default	0	0	0	0	0	0	1	1

Clock control register CKCON(0X8E) contains WD[1:0] bits select Watchdog timer timeout period. The Watchdog is clocked directly from CLK pin, and PMM mode directly affects its timeout period. It is increased 256 times when CS8959 is in PMM mode. This allows the watchdog period to remain synchronized with device operation.

The Watchdog has four timeout selections based on the input CLK clock frequency as shown in the figure. The selections are a pre-selected number of clocks. Therefore, the actual timeout interval is dependent on the CLK frequency.

Watchdog Intervals

WD[1:0]	Watchdog Interval	Number of clocks
00	2^{17}	131072
01	2^{20}	1048576
10	2^{23}	8388608
11	2^{26}	67108864

Note that the time period shown above is for the interrupt events. The RESET, when enabled, will occur 512 clocks later regardless of whether the interrupt is used. Therefore the actual Watchdog timeout period is the number shown above plus 512 clocks (always CLK pin).

20. RTC OPERATION

20.1 REAL-TIME CLOCK

The CS8959 has built-in simple real-time clock to keep track of time stamp of seconds, minutes and hours. It operates on the low-frequency 32.768KHz reference clock crystal. No day information will be kept. If this feature is needed, the firmware engineer needs to write firmware to keep track of the day changes and use XRAM to store the date information. No alarm, watchdog timer or minute stopwatch hardware is implemented at this moment.

The programmer needs to set at least one of the interrupt enable bits in order for the RTC function to work properly. The interrupt will happen at interrupt 2.

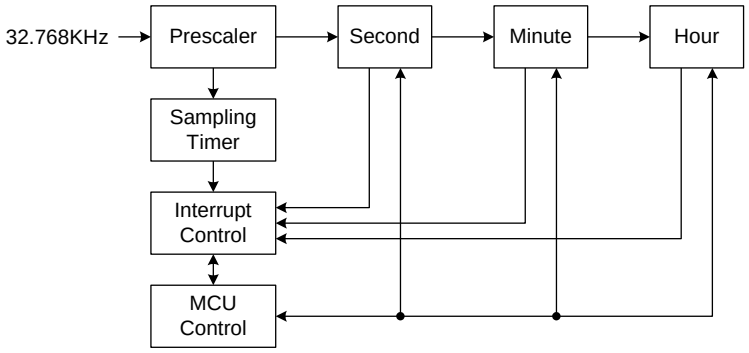


Figure-55. RTC Configuration

20.2 RTC Registers Configure

RTC Control Register (RTCCON)

RTCCON(w/r) : (XFR 0X0FE0) RTC Control Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	RTC_A4	RTC_A3	RTC_A2	RTC_A1	RTC_A0	RTC_WR	-	RTC_EN
Default	0	0	0	0	0	0	-	0

RTC_A4-0 - The RTC register address to operate.
RTC_WR - RTC_WR signal
1: Write 1 to set the RTC_WR signal to high. The firmware needs to put the data at RTCDAT register and write this bit to high for at least 200ns to ensure correct writing.
0: Write 0 to clear the RTC_WR signal to 0. Make it inactive.
RTC_EN - RTC enable
1: Write 1 to enable the RTC output. (connect to RTC OE)
0: Write 0 to disable the RTC output and set all output to 0.

RTC Data Register (RTCDAT)

RTCDAT(w/r) : (XFR 0X0FE1) RTC Data Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	RTC_D7	RTC_D6	RTC_D5	RTC_D4	RTC_D3	RTC_D2	RTC_D1	RTC_D0
Default	0	0	0	0	0	0	0	0

When write, this byte stores the byte to be written. When read, this byte presents the data coming from RTC.

Example: If you want to configure RTC Minute Register (Address :5H) to 0X45 and enable RTC output, you can use the following instructions*:*

```
MOV DPTR, #0FE0H;
MOV A, #2DH;
MOVX @DPTR, A;
MOV DPTR, #0FE1H;
MOV A, #45H;
MOVX @DPTR, A ;
```

20.3 Sampling Timer

The sampling timer is designed to support application software. You can choose a frequency from SAMx bits of the RTCIEN1 register. The sampling timer will generate an interrupt based on that frequency. You can use this timer for digitizer sampling, keyboard de-bouncing, or communication polling in your application. The sampling timer only operates if the RTC is enabled. The predefined frequency is shown below.

Frequency Choose SAMS

SAM7	512Hz
SAM6	256Hz
SAM5	128Hz
SAM4	64Hz
SAM3	32Hz
SAM2	16Hz
SAM1	8Hz
SAM0	4Hz

20.4 RTC Register Definitions

Note that for each writing activity, the firmware needs to set RTCCON[2] to 1 and clear it to 0 to generate a write clock pulse. Setting RTCCON[1] to 1 will enable the RTC module to output specified address data. Otherwise the data read in RTCDAT will be 0.

To avoid accidentally writing data to these registers during system power on/off, a safe-guard scheme is added. All the RTC registers except RTCIEN1, RTCIEN2, RTCSTS1 and RTCSTS2, are protected with this approach. Before writing data into these protected registers, the firmware had to follow a special sequence in order to enable writing these spare registers.

1. Write 1 to bit 7 of RTCIEN2,
2. Write 0 to bit 7 of RTCSTS2,
3. Write 0 to bit 7 RTCIEN2,
4. Write 1 to bit 7 of RTCSTS2.

After enabling the permission, the firmware can write to any other RTC registers, in any order, with no restriction. And the firmware needs to write 8'h10100000 to RTCSTS2 to turn off writing spare registers immediately.

There will be 16 spare 8-bit registers residing in the RTC module. It's fine to read all the RTC spare registers at any time.

RTCIEN1(w/r) : (RTC_A4-0 : 0X00) RTC Interrupt Enable 1 Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SAM7_IE	SAM6_IE	SAM5_IE	SAM4_IE	SAM3_IE	SAM2_IE	SAM1_IE	SAM0_IE
Default	0	0	0	0	0	0	0	0

SAM7_IE =1 → Enables SAM7 interrupt
=0 → Disables SAM7 interrupt
SAM6_IE =1 → Enables SAM6 interrupt
=0 → Disables SAM6 interrupt
SAM5_IE =1 → Enables SAM5 interrupt
=0 → Disables SAM5 interrupt
SAM4_IE =1 → Enables SAM4 interrupt
=0 → Disables SAM4 interrupt
SAM3_IE =1 → Enables SAM3 interrupt
=0 → Disables SAM3 interrupt
SAM2_IE =1 → Enables SAM2 interrupt
=0 → Disables SAM2 interrupt
SAM1_IE =1 → Enables SAM1 interrupt
=0 → Disables SAM1 interrupt
SAM0_IE =1 → Enables SAM0 interrupt
=0 → Disables SAM0 interrupt

RTCIEN2(w/r) : (RTC_A4-0 : 0X01) RTC Interrupt Enable 2 Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	Unlock	-	-	ALM_IE	Day_IE	Hour_IE	Min_IE	Sec_IE
Default	0	-	-	0	0	0	0	0

Unlock : Reserved to unlock spare area written permission
ALM_IE : An alarm interrupt flag is raised when the min, hour, day register matched.
=1 → Enables this alarm interrupt
=0 → Disables this alarm interrupt
Day_IE : A day interrupt flag is raised for every 24-hour clock increment (at midnight).
=1 → Enables the day interrupt.
=0 → Disables the day interrupt.
Hour_IE : A hour interrupt flag is raised every hour.
=1 → Enables the hour interrupt.
=0 → Disables the hour interrupt.
Min_IE : A minute interrupt flag is raised every minute.
=1 → Enables the minute interrupt.
=0 → Disables the minute interrupt.
Sec_IE : A second interrupt flag is raised every second.
=1 → Enables the second interrupt.
=0 → Disables the second interrupt.

RTCSTS1(w/r) : (RTC_A4-0 : 0X02) RTC Interrupt Status 1 Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SAM7_IF	SAM6_IF	SAM5_IF	SAM4_IF	SAM3_IF	SAM2_IF	SAM1_IF	SAM0_IF
Default	0	0	0	0	0	0	0	0

SAM7_IF =1 → SAM7 interrupt occurred; Write 1 to clear.
=0 → No SAM7 interrupt occurred
SAM6_IF =1 → SAM6 interrupt occurred; Write 1 to clear.
=0 → No SAM6 interrupt occurred
SAM5_IF =1 → SAM5 interrupt occurred; Write 1 to clear.
=0 → No SAM5 interrupt occurred
SAM4_IF =1 → SAM4 interrupt occurred; Write 1 to clear.
=0 → No SAM4 interrupt occurred
SAM3_IF =1 → SAM3 interrupt occurred; Write 1 to clear.
=0 → No SAM3 interrupt occurred
SAM2_IF =1 → SAM2 interrupt occurred; Write 1 to clear.
=0 → No SAM2 interrupt occurred
SAM1_IF =1 → SAM1 interrupt occurred; Write 1 to clear.
=0 → No SAM1 interrupt occurred
SAM0_IF =1 → SAM0 interrupt occurred; Write 1 to clear.
=0 → No SAM0 interrupt occurred

RTCSTS2(w/r) : (RTC_A4-0 : 0X03) RTC Interrupt Status 2 Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	-	-	-	ALM_IF	Day_IF	Hour_IF	Min_IF	Sec_IF
Default	-	-	-	0	0	0	0	0

ALM_IF =1 → Alarm interrupt occurred. Write 1 to clear.
=0 → No alarm interrupt.
Day_IE =1 → Day interrupt occurred. Write 1 to clear.
=0 → No Day interrupt.
Hour_IE =1 → Hour interrupt occurred. Write 1 to clear.
=0 → No Hour interrupt.
Min_IE =1 → Minute interrupt occurred. Write 1 to clear.
=0 → No Minute interrupt.
Sec_IE =1 → Second interrupt occurred. Write 1 to clear.
=0 → No Second interrupt.

RTCSEC(w/r) : (RTC_A4-0 : 0X04) RTC Second Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	ADJ_EN	RUN_ADJ	ADJ_VAL5	ADJ_VAL4	ADJ_VAL3	ADJ_VAL2	ADJ_VAL1	ADJ_VAL0
Default	-	-	-	0	0	0	0	0

ADJ_EN =1 → Enable automatic time adjustment.
=0 → Disable automatic time adjustment.
RUN_ADJ =1 → RTC clock runs too fast, need to count more.
=0 → RTC clock runs too slow, need to count less.
ADJ_VAL5-0 → Write the automatic clock adjustment values (need in VBAT domain).
→ Read second counter value.

RTCMIN(w/r) : (RTC_A4-0 : 0X05) RTC Minute Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	CLR_CNT	-	MIN_CNT5	MIN_CNT4	MIN_CNT3	MIN_CNT2	MIN_CNT1	MIN_CNT0
Default	0	-	0	0	0	0	0	0

CLR_CNT → Write 1 to clear the second counter. User must write another 0 into this bit to refresh the second counter after clear by writing 1.

MIN_CNT5-0 → User can program the minute counter by writing a value.

RTCHOUR(w/r) : (RTC_A4-0 : 0X06) RTC Hour Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	AM_PM	-	-	HR_CNT4	HR_CNT3	HR_CNT2	HR_CNT1	HR_CNT0
Default	0	-	-	0	0	0	0	0

AM_PM =1 → Write 1 to enable 12 Hour AM/PM format.
=0 → Write 0 to enable 24 Hour format.

HR_CNT4 =1 → If AM_PM = 1, it means PM now. If AM_PM = 0, it is the MSB of hour counter.
=0 → If AM_PM = 1, it means AM now. If AM_PM = 0, it is the MSB of hour counter.

HR_CNT3-0 → If AM_PM = 1, HR_CNT3-0 indicate the hour value 0 to 11.
→ If AM_PM = 0, HR_CNT4-0 indicate the hour value 0 to 23.

RTCDAYL(w/r) : (RTC_A4-0 : 0X07) RTC Day Counter Low Byte Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	Day counter starting from Jan. 1, year 2000, this is the low byte “RTCDAY[7:0]”							
Default	0X00							

RTCDAYH(w/r) : (RTC_A4-0 : 0X08) RTC Day Counter High Byte Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	Day counter starting from Jan. 1, year 2000, this is the high byte “RTCDAY[15:8]”							
Default	0X00							

ALMSEC(w/r) : (RTC_A4-0 : 0X09) RTC Alarm Second Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	Reserved	Reserved	ALM_SEC5	ALM_SEC4	ALM_SEC3	ALM_SEC2	ALM_SEC1	ALM_SEC0
Default	-	-	0	0	0	0	0	0

ALM_SEC5-0 → Expected alarm “second” setting.

ALMMIN(w/r) : (RTC_A4-0 : 0X0A) RTC Alarm Minute Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	Reserved	Reserved	ALM_MIN5	ALM_MIN4	ALM_MIN3	ALM_MIN2	ALM_MIN1	ALM_MIN0
Default	-	-	0	0	0	0	0	0

ALM_MIN5-0 → Expected alarm “minute” setting.

ALMHOUR(w/r) : (RTC_A4-0 : 0X0B) RTC Alarm Hour Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	Reserved	Reserved	Reserved	ALM_HOUR4	ALM_HOUR3	ALM_HOUR2	ALM_HOUR1	ALM_HOUR0
Default	-	-	-	0	0	0	0	0

ALM_HOUR4-0 → Expected alarm “hour” settings, in 24-hour representation.

ALMDAYL(w/r) : (RTC_A4-0 : 0X0C) RTC Alarm Day Low Byte Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	Expected alarm “day” settings. Day counter starting from Jan. 1, year 2000, this is the high byte.							
Default	0X00							

ALMDAYL → Expected alarm “day” settings. Day counter starting from Jan. 1, year 2000, this is the low byte.

ALMDAYH(w/r) : (RTC_A4-0 : 0X0D) RTC Alarm Day High Byte Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	Expected alarm “day” settings. Day counter starting from year 2000, this is the low byte							
Default	0X00							

ALMDAYH → Expected alarm “day” settings. Day counter starting from Jan. 1, year 2000, this is the high byte.

Note that the actual day, month, and year shall be calculated by firmware. The hardware only provides the day counter after Jan. 1, 2000.

Address 10 – 1FH: RTC Spare Register 0 - 15 (RTCSP1 – RTCSP15)
These spare registers are in RTC power domain. The data could be kept as long as VBAT is available

21. PWM DAC

Each output pulse width of PWM DAC converter is controlled by an 8-bit register in XFR. The frequency of PWM clock is fosc/1024 or fosc/512, selected by PWMF. And the total duty cycle step of these DAC outputs is 253 or 256, selected by DIV253. If DIV253=1, writing FDH/FEH/FFH to DAC register generates stable high output. If DIV253=0, the output pulses low at least once even if the DAC register's content is FFH. Writing 00H to DAC register generates stable low output.

DAC Registers List

DA0	0X0F20(r/w)	Pulse width of PWM DAC 0							
DA1	0X0F21(r/w)	Pulse width of PWM DAC 1							
DA2	0X0F22(r/w)	Pulse width of PWM DAC 2							
DA3	0X0F23(r/w)	Pulse width of PWM DAC 3							
DA4	0X0F24(r/w)	Pulse width of PWM DAC 4							
DA5	0X0F25(r/w)	Pulse width of PWM DAC 5							
DA6	0X0F26(r/w)	Pulse width of PWM DAC 6							
DA7	0X0F27(r/w)	Pulse width of PWM DAC 7							
DA8	0X0F28(r/w)	Pulse width of PWM DAC 8							
RESERVED	0X0F29(r/w)	-	-	-	-	-	-	-	-
RESERVED	0X0F2A(r/w)	-	-	-	-	-	-	-	-
PFC	0X0F2B(r/w)	PFC13	PFC12	PFC11	PFC10	PFC03	PFC02	PFC01	PFC00
PADMOD	0X0F50(w)								DA8E
PADMOD	0X0F51(w)	DA7E	DA6E	DA5E	DA4E	DA3E	DA2E	DA1E	DA0E
OPTION	0X0F56(r/w)	PWMIF	DIV253	-	ENSCL1	ENSCL0	-	-	-

OPTION (r/w) : (XFR 0X0F56) Chip Option Configuration (default 0X00)

	Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Name	PWMF	DIV253	-	ENSCL1	ENSCL0	-	-	-
Default	0	0	-	0	0	-	-	-

PWMF = 1 → Selects system clock/512 as PWM clock frequency.
= 0 → Selects system clock/1024 as PWM clock frequency.
DIV253 = 1 → PWM pulse width is 253-step resolution.
= 0 → PWM pulse width is 256-step resolution.

DA0 (r/w) : (XFR 0X0F20) DA0 Pulse Width Register (default 0X80)

	Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Name	DA0[7:0]							
Default	0X80							

DA0[7:0] → The output pulse width control for PWM DAC0.

DA1 (r/w) : (XFR 0X0F21) DA1 Low Pulse Width Register (default 0X80)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DA1[7:0]							
Default	0X80							

DA1[7:0] → The output pulse width control for PWM DAC1.

DA2 (r/w) : (XFR 0X0F22) DA2 Pulse Width Register (default 0X80)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DA2[7:0]							
Default	0X80							

DA2[7:0] → The output pulse width control for PWM DAC2.

DA3 (r/w) : (XFR 0X0F23) DA3 Pulse Width Register (default 0X80)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DA3[7:0]							
Default	0X80							

DA3[7:0] → The output pulse width control for PWM DAC3.

DA4 (r/w) : (XFR 0X0F24) DA4 Pulse Width Register (default 0X80)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DA4[7:0]							
Default	0X80							

DA4[7:0] → The output pulse width control for PWM DAC4.

DA5 (r/w) : (XFR 0X0F25) DA5 Pulse Width Register (default 0X80)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DA5[7:0]							
Default	0X80							

DA5[7:0] → The output pulse width control for PWM DAC5.

DA6 (r/w) : (XFR 0X0F26) DA6 Pulse Width Register (default 0X80)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DA6[7:0]							
Default	0X80							

DA6[7:0] → The output pulse width control for PWM DAC6.

DA7 (r/w) : (XFR 0X0F27) DA7 Pulse Width Register (default 0X80)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DA7[7:0]							
Default	0X80							

DA7[7:0] → The output pulse width control for PWM DAC7.

DA8 (r/w) : (XFR 0X0F28) DA8 Pulse Width Register (default 0X80)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DA8[7:0]							
Default	0X80							

DA8[7:0] → The output pulse width control for PWM DAC8.

PFC (r/w) : (XFR 0X0F2B) Pre-frequency Scale Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	PFC13	PFC12	PFC11	PFC10	PFC03	PFC02	PFC01	PFC00
Default	0	0	0	0	0	0	0	0

PFC → Output period control for DA0, DA1.
DA1 period = system clock period X $2^{(PFC13, PFC12, PFC11, PFC10)}$. The minimum value for DA1 period is system clock period x 2
DA0 period = system clock period X $2^{(PFC03, PFC02, PFC01, PFC00)}$. The minimum value for DA0 period is system clock period x 2
All of PWM DAC converters are centered with value 0X80 after power on.

PADMOD0 (r/w): (XFR 0X0F50) Pad mode control registers 0. (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SPI_MOD	-	-	-	-	-	-	DA8E
Default	0	-	-	-	-	-	-	0

SPI_MOD = 1 → Pin "P7.2/ET2T/SCS" is SPI_SCS; pin "P7.3/ET2ET/MOSI" is SPI_MOSI.
→ Pin "P7.4/ET3T/MISO" is SPI_MISO; pin "P7.5/ET3ET/MSCK" is SPI_MSCK.
= 0 → Pin "P7.2/ET2T/SCS" is P7.2; pin "P7.3/ET2ET/MOSI" is P7.3.
→ Pin "P7.4/ET3T/MISO" is P7.4; pin "P7.5/ET3ET/MSCK" is P7.5.

DA8E = 1 → Pin "P6.4/PWM8" is PWM8. Select DA8 output
= 0 → Pin "P6.4/PWM8" is P6.4.

PADMOD1 (r/w) : (XFR 0X0F51) Pad mode control registers 1. (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	DA7E	DA6E	DA5E	DA4E	DA3E	DA2E	DA1E	DA0E
Default	0	0	0	0	0	0	0	0

DA7E = 1 → Pin "P9.3/PWM7" is PWM7. Select DA7 output
= 0 → Pin "P9.3/PWM7" is P9.3.
DA6E = 1 → Pin "P6.3/PWM6" is PWM6. Select DA6 output
= 0 → Pin "P6.3/PWM6" is P6.3.
DA5E = 1 → Pin "P9.2/PWM5" is PWM5. Select DA5 output
= 0 → Pin "P9.2/PWM5" is P9.2.
DA4E = 1 → Pin "P6.2/PWM4" is PWM4. Select DA4 output
= 0 → Pin "P6.2/PWM4" is P6.2.
DA3E = 1 → Pin "P9.1/PWM3" is PWM3. Select DA3 output
= 0 → Pin "P9.1/PWM3" is P9.1.
DA2E = 1 → Pin "P6.1/PWM2" is PWM2. Select DA2 output
= 0 → Pin "P6.1/PWM2" is P6.1.
DA1E = 1 → Pin "P9.0/PWM1" is PWM1. Select DA1 output
= 0 → Pin "P9.0/PWM1" is P9.0.
DA0E = 1 → Pin "P6.0/PWM0" is PWM0. Select DA0 output
= 0 → Pin "P6.0/PWM0" is P6.0.

22. LOW VOLTAGE DETECTION

The Low Voltage Detection Circuit (LVDT) detects the condition of VDD5 supply and V25 supply. When enabled, it detect VDD5 < VTH condition, and can be configured to generate an interrupt (LVDT_INT) or a system reset (LVDT_RST). Because the existence of on-chip regulator the system and user program should have ample time to responds to the interrupt. The larger the decoupling capacitance on V25, the longer this time can be extended. To ensure a reliable MCU operation and prevent loss of data, user program should terminate all critical process and wait for power supply to get back to normal or as power supply continue to deteriorate and finally reset the system.

The block diagram of Low Voltage Detection Circuit is shown in the following figure.

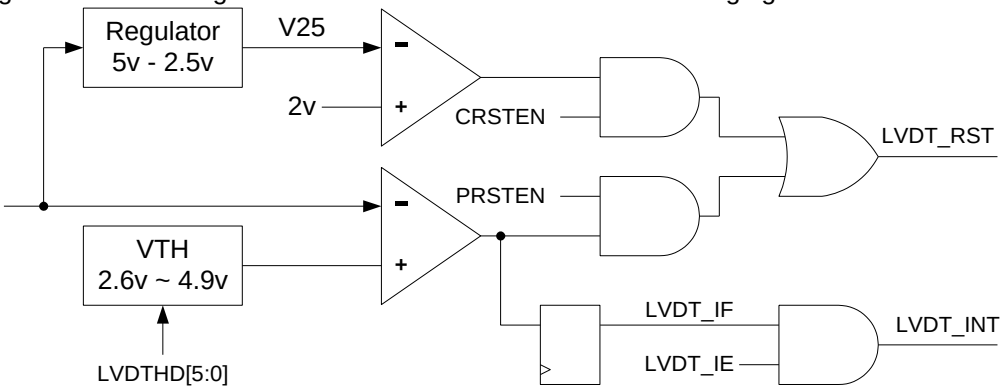


Figure-56. LVDT Diagram

The default state of the LVDT circuit for V25 is enabled and the LVDT circuit for VDD5 is disabled. The user program needs to enable the preferred configuration and set the appropriate detection threshold level. The following XFR registers are used for this purpose.

POWER(w/r) : (XFR 0X0F9E) Power management (default 0X02)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	Reserved	Reserved	-	-	-	PRSTEN	CRSTEN	-
Default	0	0	-	-	-	0	1	-

PRSTEN = 1 → Enable LVDT for power supply VDD5 (5v).
= 0 → Disable LVDT for power supply VDD5 (5v).
CRSTEN = 1 → Enable LVDT for core power V25 (2.5v).
= 0 → Disable LVDT for core power V25 (2.5v).

LVDTCTL(w/r) : (SFR 0X9F) LVDT Control Register (default 0X20)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	LVDT_IF	LVDT_IE	LVDTHD[5:0]					
Default	0	0	0X20					

LVDT_IF → This bit is set when VDD5 lower than the threshold level and the LVDT circuit will generate an interrupt (LVDT_INT) to inform system if LVDT_IE is set. Writing a zero when VDD5 higher than the threshold level will clear this bit.
LVDT_IE = 1 → Enable LVDT interrupt.
= 0 → Disable LVDT interrupt.
LVDTHD[5:0] → The threshold level for VDD5. LVDTHD = 0X3F will set the detection threshold at its minimum (approximately 2.6V), and LVDTHD = 0X00 will set the detection threshold at its maximum (approximately 4.9V).

23. ECT

The CS8959 ECT (Enhanced Capture Timer) module has the features of the Time2. But ECT module does not be a baud rate generator. CS8959 has 6 ECTs for various frequencies detection.

23.1 ECT0

ECT0 is a 16-bit Timer/Counter, which can operate as either an event timer or an event counter, as selected by C/ECT0T in the special function register T2CON0 (0XA9). ECT0 has two operating modes: Capture Mode and Auto-reload Mode selected by bits in the T2CON0.

T2CON0 (r/w) : (SFR 0XA9) ECT0 Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	E0_TF2	E0_EXF2	E0_TF2EN	E0_EXF2EN	E0_EXEN2	E0_TR2	C/ECT0T	E0_CP/RL2
Default	0	0	0	0	0	0	0	0

E0_TF2 : ECT0 interrupt (overflow) flag. Must be cleared by software.
E0_EXF2 : Falling edge indicator on ECT0ET pin when E0_EXEN2=1. Must be cleared by software.
E0_TF2EN : Overflow interrupt enable.
=1 → Enable overflow interrupt
=0 → Disable overflow interrupt
E0_EXF2EN : Falling edge interrupt enable
=1 → Enable falling edge interrupt
=0 → Disable falling edge interrupt
E0_EXEN2 : Enable ECT0ET pin functionality.
=1 → Allows capture or reload as a result of ECT0ET pin falling edge
=0 → Ignore ECT0ET events
E0_TR2 : Start/stop ECT0
=1 → Start
=0 → Stop
C/ECT0T : Timer/counter select
=1 → External event counter. Clock source is ECT0T pin
=0 → Timer Internally clocked
E0_CP/RL2 : Capture/reload select
=1 → ECT0ET pin falling edge causes capture to occur when E0_EXEN2=1
=0 → Automatic reload occurs: on ECT0 overflow or falling edge of ECT0ET pin when E0_EXEN2=1.

23.1.1 Capture Mode

In the capture mode, there are two options that are selected by bit E0_EXEN2 in T2CON0. If E0_EXEN2=0, then ECT0 is a 16-bit timer or counter (as selected by C/ECT0T in T2CON0) which, upon overflowing sets bit E0_TF2, the ECT0 overflow bit. This bit can be used to generate an interrupt (by enabling the ECT0 interrupt bit in the EXINTEN register). If E0_EXEN2= 1, ECT0 operates as described above, but with the added feature that a 1- to -0 transition at external input ECT0ET causes the current value in the ECT0 registers, TL2_0(0XAC) and TH2_0(0XAD), to be captured into registers RLDL_0(0XAA) and RLDH_0(0XAB), respectively. In addition, the transition at ECT0ET causes bit E0_EXF2 in T2CON0 to be set, and E0_EXF2 like E0_TF2 can generate an interrupt (which vectors to the same location as ECT0 overflow interrupt.

The ECT0 interrupt service routine can interrogate E0_TF2 and E0_EXF2 to determine which event caused the interrupt). The capture mode is illustrated in Figure (There is no reload value for TL2_0 and TH2_0 in this mode. Even when a capture event occurs from ECT0ET, the counter keeps on counting ECT0ET pin transitions or osc/12 pulses.)

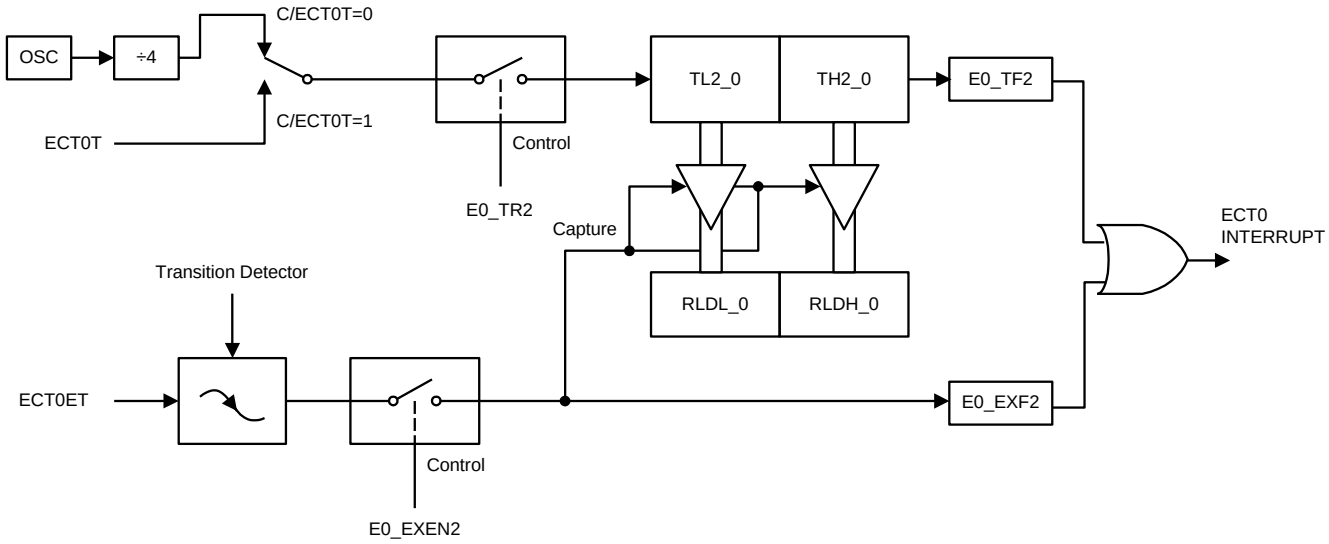


Figure-57. ECT0 Capture Mode

23.1.2 Auto-Reload Mode

In the 16-bit auto-reload mode, ECT0 can be configured (as either a timer or counter (C/ECT0T in T2CON0)). ECT0 will count up automatically. In this mode there are two options selected by bit E0_EXEN2 in T2CON0 register. If E0_EXEN2=0, then ECT0 counts up to 0XFFFF and sets the E0_TF2 (Overflow Flag) bit upon overflow. This causes the ECT0 registers to be reloaded with the 16-bit value in RLDL_0 and RLDH_0. The values in RLDL_0 and RLDH_0 are preset by software means. If E0_EXEN2=1, then a 16-bit reload can be triggered either by an overflow or by a 1-to-0 transition at input ECT0ET. This transition also sets the E0_EXF2 bit. The ECT0 interrupt, if enabled, can be generated when either E0_TF2 or E0_EXF2 are 1.

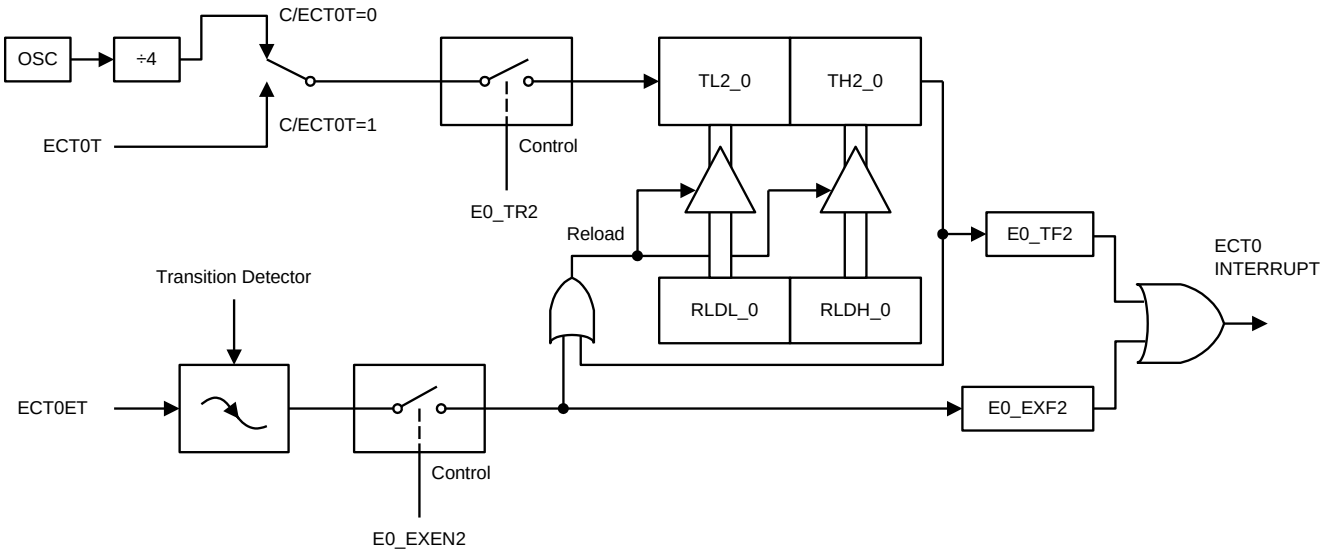


Figure-58. ECT0 Auto-Reloaded Mode

23.2 ECT1

ECT1 is a 16-bit Timer/Counter, which can operate as either an event timer or an event counter, as selected by C/ECT1T* in the special function register T2CON1 (0XB1). ECT1 has two operating modes: Capture, Auto-reload, which are selected by bits in the T2CON1.

T2CON1 (r/w) : (SFR 0XB1) ECT1 Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	E1_TF2	E1_EXF2	E1_TF2EN	E1_EXF2EN	E1_EXEN2	E1_TR2	C/ECT1T	E1_CP/RL2
Default	0	0	0	0	0	0	0	0

E1_TF2 : ECT1 interrupt (overflow) flag. Must be cleared by software.
E1_EXF2 : Falling edge indicator on ECT1ET pin when E1_EXEN2=1. Must be cleared by software.
E1_TF2EN : Overflow interrupt enable.
=1 → Enable overflow interrupt
=0 → Disable overflow interrupt
E1_EXF2EN : Falling edge interrupt enable
=1 → Enable falling edge interrupt
=0 → Disable falling edge interrupt
E1_EXEN2 : Enable ECT1ET pin functionality.
=1 → Allows capture or reload as a result of ECT1ET pin falling edge
=0 → Ignore ECT1ET events
E1_TR2 : Start/stop ECT1
=1 → Start
=0 → Stop
C/ECT1T : Timer/counter select
=1 → External event counter. Clock source is ECT1T pin
=0 → Timer Internally clocked
E1_CP/RL2 : Capture/reload select
=1 → ECT1ET pin falling edge causes capture to occur when E1_EXEN2=1
=0 → Automatic reload occurs: on ECT1 overflow or falling edge of ECT1ET pin when E1_EXEN2=1.

23.2.1 Capture Mode

In the capture mode, there are two options that are selected by bit E1_EXEN2 in T2CON1. If E1_EXEN2=0, then ECT1 is a 16-bit timer or counter (as selected by C/ECT1T in T2CON1) which, upon overflowing sets bit E1_TF2, the ECT1 overflow bit. This bit can be used to generate an interrupt (by enabling the ECT1 interrupt bit in the EXINTEN register). If E1_EXEN2= 1, ECT1 operates as described above, but with the added feature that a 1- to -0 transition at external input ECT1ET causes the current value in the ECT1 registers, TL2_1(0XB4) and TH2_1(0XB5), to be captured into registers RLDL_1(0XB2) and RLDH_1(0XB3), respectively. In addition, the transition at ECT1ET causes bit E1_EXF2 in T2CON1 to be set, and E1_EXF2 like E1_TF2 can generate an interrupt (which vectors to the same location as ECT1 overflow interrupt.

The ECT1 interrupt service routine can interrogate E1_TF2 and E1_EXF2 to determine which event caused the interrupt). The capture mode is illustrated in Figure (There is no reload value for TL2_1 and TH2_1 in this mode. Even when a capture event occurs from ECT1ET, the counter keeps on counting ECT1ET pin transitions or osc/12 pulses.).

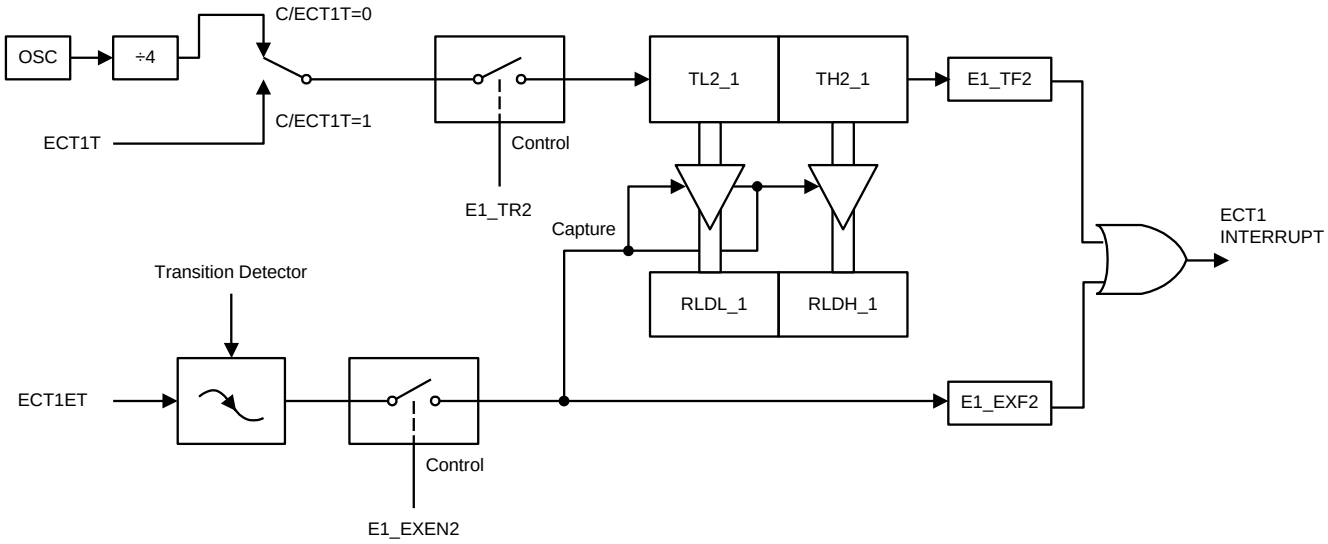


Figure-59. ECT1 Capture Mode

23.2.2 Auto-Reload Mode

In the 16-bit auto-reload mode, ECT1 can be configured (as either a timer or counter (C/ECT1T in T2CON1)). ECT1 will count up automatically. In this mode there are two options selected by bit E1_EXEN2 in T2CON1 register. If E1_EXEN2=0, then ECT1 counts up to 0XFFFF and sets the E1_TF2 (Overflow Flag) bit upon overflow. This causes the ECT1 registers to be reloaded with the 16-bit value in RLDL_1 and RLDH_1. The values in RLDL_1 and RLDH_1 are preset by software means. If E1_EXEN2=1, then a 16-bit reload can be triggered either by an overflow or by a 1-to-0 transition at input ECT1ET. This transition also sets the E1_EXF2 bit. The ECT1 interrupt, if enabled, can be generated when either E1_TF2 or E1_EXF2 are 1.

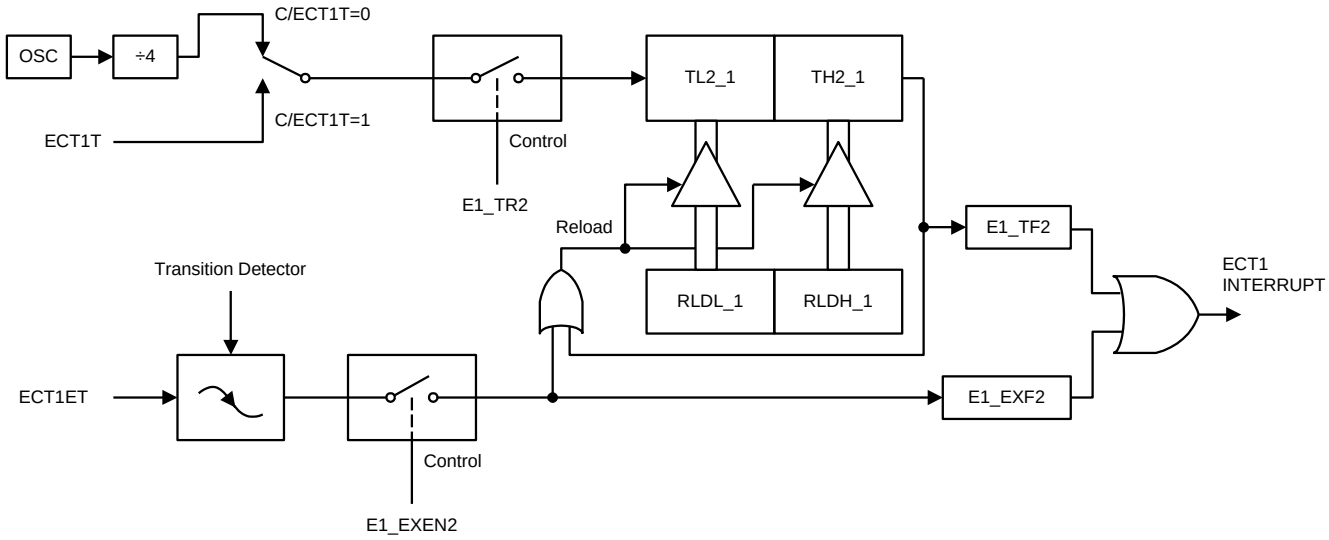


Figure-60. ECT1 Auto-Reloaded Mode

23.3 ECT2

ECT2 is a 16-bit Timer/Counter, which can operate as either an event timer or an event counter, as selected by C/ECT2T* in the special function register T2CON2 (0XB9). ECT2 has two operating modes: Capture, Auto-reload, which are selected by bits in the T2CON2.

ECT2 Configuration Register

T2CON2 (r/w) : (SFR 0XB9) ECT2 Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	E2_TF2	E2_EXF2	E2_TF2EN	E2_EXF2EN	E2_EXEN2	E2_TR2	C/ECT2T	E2_CP/RL2
Default	0	0	0	0	0	0	0	0

E2_TF2 : ECT2 interrupt (overflow) flag. Must be cleared by software.
E2_EXF2 : Falling edge indicator on ECT2ET pin when E2_EXEN2=1. Must be cleared by software.
E2_TF2EN : Overflow interrupt enable.
=1 → Enable overflow interrupt
=0 → Disable overflow interrupt
E2_EXF2EN : Falling edge interrupt enable
=1 → Enable falling edge interrupt
=0 → Disable falling edge interrupt
E2_EXEN2 : Enable ECT2ET pin functionality.
=1 → Allows capture or reload as a result of ECT2ET pin falling edge
=0 → Ignore ECT2ET events
E2_TR2 : Start/stop ECT2
=1 → Start
=0 → Stop
C/ECT2T : Timer/counter select
=1 → External event counter. Clock source is ECT2T pin
=0 → Timer Internally clocked
E2_CP/RL2 : Capture/reload select
=1 → ECT2ET pin falling edge causes capture to occur when E2_EXEN2=1
=0 → Automatic reload occurs: on ECT1 overflow or falling edge of ECT2ET pin when E2_EXEN2=1.

23.3.1 Capture Mode

In the capture mode there are two options that are selected by bit E2_EXEN2 in T2CON2. If E2_EXEN2=0, then ECT2 is a 16-bit timer or counter (as selected by C/ECT2T in T2CON2) which, upon overflowing sets bit E2_TF2, the ECT2 overflow bit. This bit can be used to generate an interrupt (by enabling the ECT2 interrupt bit in the EXINTEN register). If E2_EXEN2= 1, ECT2 operates as described above, but with the added feature that a 1- to -0 transition at external input ECT2ET causes the current value in the ECT2 registers, TL2_2(0XBC) and TH2_2(0XBD), to be captured into registers RLDL_2(0XBA) and RLDH_2(0XBB), respectively. In addition, the transition at ECT2ET causes bit E2_EXF2 in T2CON2 to be set, and E2_EXF2 like E2_TF2 can generate an interrupt (which vectors to the same location as ECT2 overflow interrupt.

The ECT2 interrupt service routine can interrogate E2_TF2 and E2_EXF2 to determine which event caused the interrupt). The capture mode is illustrated in Figure (There is no reload value for TL2_2 and TH2_2 in this mode. Even when a capture event occurs from ECT2ET, the counter keeps on counting ECT2ET pin transitions or osc/12 pulses.)

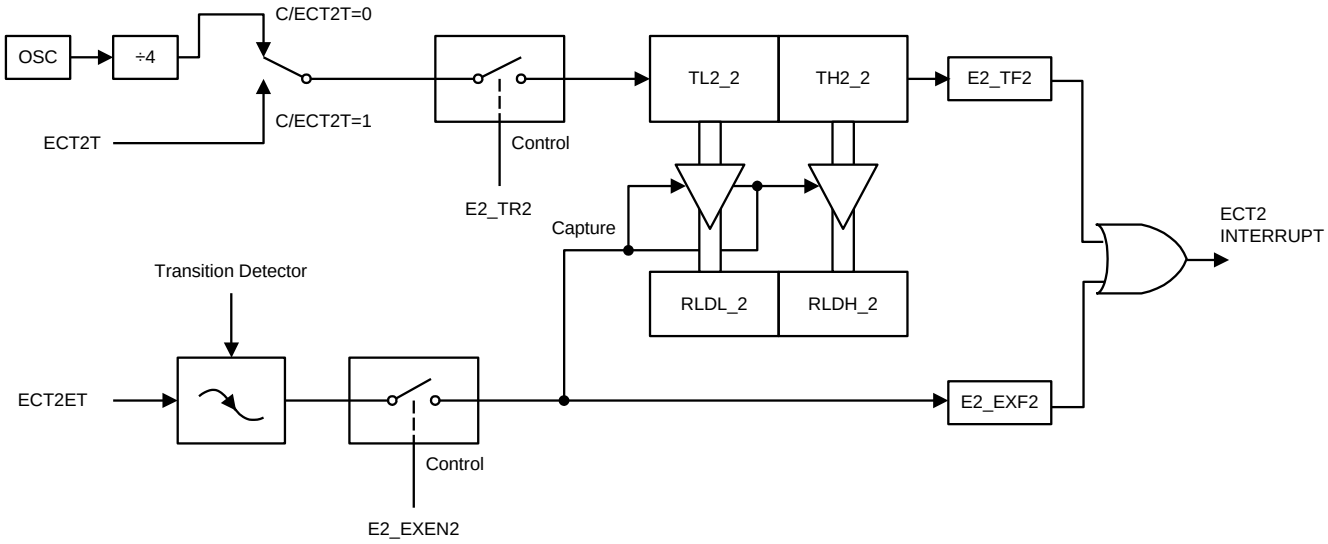


Figure-61. ECT2 Capture Mode

23.3.2 Auto-Reload Mode

In the 16-bit auto-reload mode, ECT2 can be configured (as either a timer or counter (C/ECT2T in T2CON2)). ECT2 will count up automatically. In this mode there are two options selected by bit E2_EXEN2 in T2CON2 register. If E2_EXEN2=0, then ECT2 counts up to 0XFFFF and sets the E2_TF2 (Overflow Flag) bit upon overflow. This causes the ECT2 registers to be reloaded with the 16-bit value in RLDL_2 and RLDH_2. The values in RLDL_2 and RLDH_2 are preset by software means. If E2_EXEN2=1, then a 16-bit reload can be triggered either by an overflow or by a 1-to-0 transition at input ECT2ET. This transition also sets the E2_EXF2 bit. The ECT2 interrupt, if enabled, can be generated when either E2_TF2 or E2_EXF2 are 1.

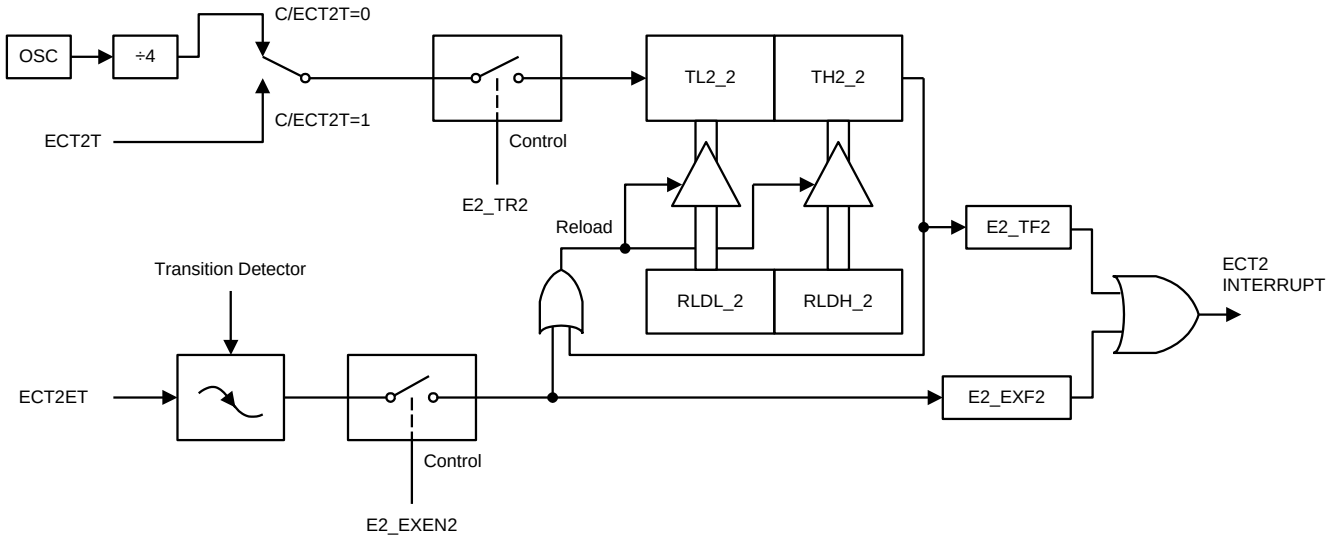


Figure-62. ECT2 Auto-Reloaded Mode

23.4 ECT3

ECT3 is a 16-bit Timer/Counter, which can operate as either an event timer or an event counter, as selected by C/ECT3T* in the special function register T2CON3 (0XD1). ECT3 has two operating modes: Capture, Auto-reload, which are selected by bits in the T2CON3.

T2CON3 (r/w) : (SFR 0XD1) ECT3 Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	E3_TF2	E3_EXF2	E3_TF2EN	E3_EXF2EN	E3_EXEN2	E3_TR2	C/ECT3T	E3_CP/RL2
Default	0	0	0	0	0	0	0	0

E3_TF2 : ECT3 interrupt (overflow) flag. Must be cleared by software.
E3_EXF2 : Falling edge indicator on ECT3ET pin when E3_EXEN2=1. Must be cleared by software.
E3_TF2EN : Overflow interrupt enable.
=1 → Enable overflow interrupt
=0 → Disable overflow interrupt
E3_EXF2EN : Falling edge interrupt enable
=1 → Enable falling edge interrupt
=0 → Disable falling edge interrupt
E3_EXEN2 : Enable ECT3ET pin functionality.
=1 → Allows capture or reload as a result of ECT3ET pin falling edge
=0 → Ignore ECT3ET events
E3_TR2 : Start/stop ECT3
=1 → Start
=0 → Stop
C/ECT3T : Timer/counter select
=1 → External event counter. Clock source is ECT3T pin
=0 → Timer Internally clocked
E3_CP/RL2 : Capture/reload select
=1 → ECT3ET pin falling edge causes capture to occur when E3_EXEN2=1
=0 → Automatic reload occurs: on ECT3 overflow or falling edge of ECT3ET pin when E3_EXEN2=1.

23.4.1 Capture Mode

In the capture mode there are two options that are selected by bit E3_EXEN2 in T2CON3. If E3_EXEN2=0, then ECT3 is a 16-bit timer or counter (as selected by C/ECT3T in T2CON3) which, upon overflowing sets bit E3_TF2, the ECT3 overflow bit. This bit can be used to generate an interrupt (by enabling the ECT3 interrupt bit in the EXINTEN register). If E3_EXEN2= 1, ECT3 operates as described above, but with the added feature that a 1- to -0 transition at external input ECT3ET causes the current value in the ECT3 registers, TL2_3(0XD4) and TH2_3(0XD5), to be captured into registers RLDL_3(0XD2) and RLDH_3(0XD3), respectively. In addition, the transition at ECT3ET causes bit E3_EXF2 in T2CON3 to be set, and E3_EXF2 like E3_TF2 can generate an interrupt (which vectors to the same location as ECT3 overflow interrupt.

The ECT3 interrupt service routine can interrogate E3_TF2 and E3_EXF2 to determine which event caused the interrupt). The capture mode is illustrated in Figure (There is no reload value for TL2_3 and TH2_3 in this mode. Even when a capture event occurs from ECT3ET, the counter keeps on counting ECT3ET pin transitions or osc/12 pulses.)

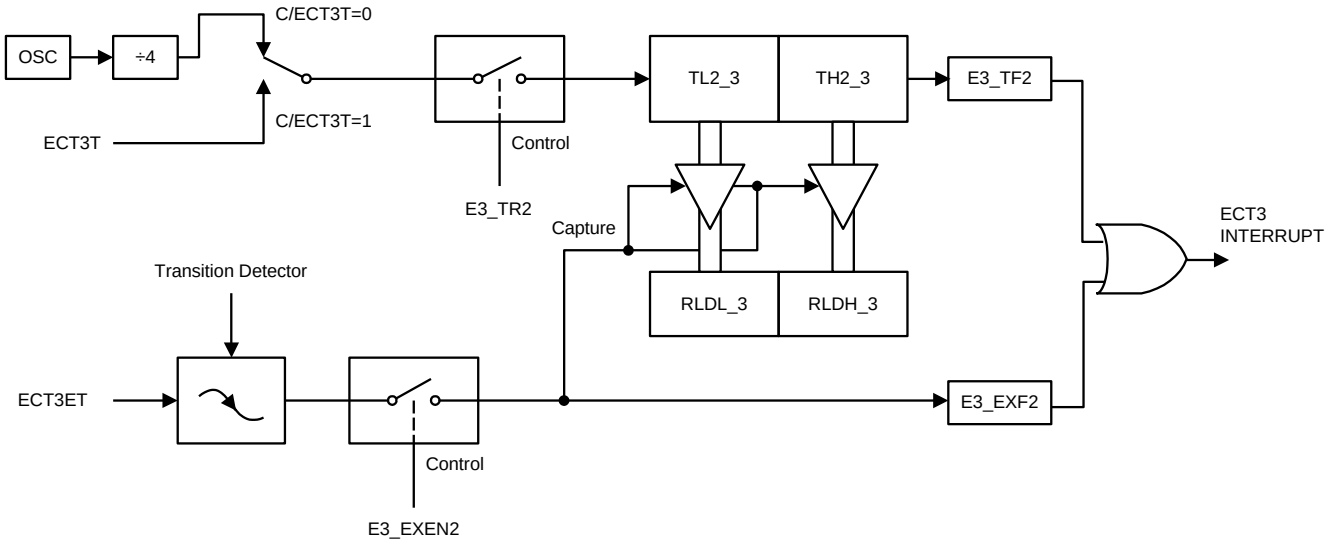


Figure-63. ECT3 Capture Mode

23.4.2 Auto-Reload Mode

In the 16-bit auto-reload mode, ECT3 can be configured (as either a timer or counter (C/ECT3T in T2CON3)). ECT3 will count up automatically. In this mode there are two options selected by bit E3_EXEN2 in T2CON3 register. If E3_EXEN2=0, then ECT3 counts up to 0XFFFF and sets the E3_TF2 (Overflow Flag) bit upon overflow. This causes the ECT3 registers to be reloaded with the 16-bit value in RLDL_3 and RLDH_3. The values in RLDL_3 and RLDH_3 are preset by software means. If E3_EXEN2=1, then a 16-bit reload can be triggered either by an overflow or by a 1-to-0 transition at input ECT3ET. This transition also sets the E3_EXF2 bit. The ECT3 interrupt, if enabled, can be generated when either E3_TF2 or E3_EXF2 are 1.

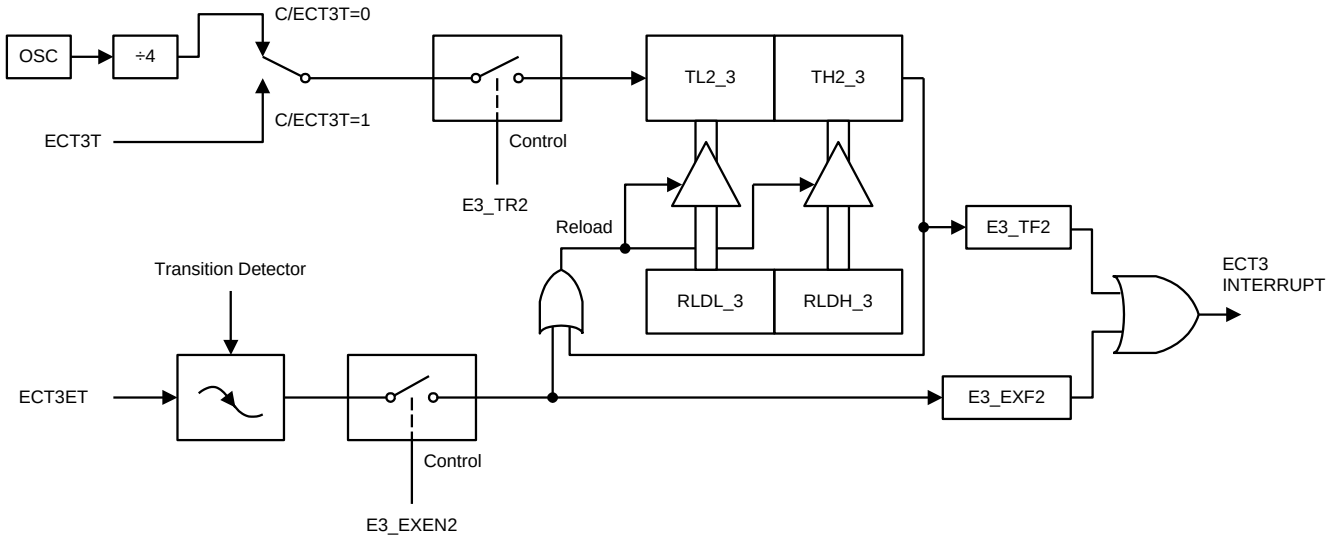


Figure-64. ECT3 Auto-Reloaded Mode

23.5 ECT4

ECT4 is a 16-bit Timer/Counter, which can operate as either an event timer or an event counter, as selected by C/ECT4T* in the special function register T2CON4 (0XD9). ECT4 has two operating modes: Capture, Auto-reload, which are selected by bits in the T2CON4.

T2CON4 (r/w) : (SFR 0XD9) ECT4 Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	E4_TF2	E4_EXF2	E4_TF2EN	E4_EXF2EN	E4_EXEN2	E4_TR2	C/ECT4T	E4_CP/RL2
Default	0	0	0	0	0	0	0	0

E4_TF2 : ECT4 interrupt (overflow) flag. Must be cleared by software.
E4_EXF2 : Falling edge indicator on ECT4ET pin when E4_EXEN2=1. Must be cleared by software.
E4_TF2EN : Overflow interrupt enable.
=1 → Enable overflow interrupt
=0 → Disable overflow interrupt
E4_EXF2EN : Falling edge interrupt enable
=1 → Enable falling edge interrupt
=0 → Disable falling edge interrupt
E4_EXEN2 : Enable ECT4ET pin functionality.
=1 → Allows capture or reload as a result of ECT4ET pin falling edge
=0 → Ignore ECT4ET events
E4_TR2 : Start/stop ECT4
=1 → Start
=0 → Stop
C/ECT4T : Timer/counter select
=1 → External event counter. Clock source is ECT4T pin
=0 → Timer Internally clocked
E4_CP/RL2 : Capture/reload select
=1 → ECT4ET pin falling edge causes capture to occur when E4_EXEN2=1
=0 → Automatic reload occurs: on ECT4 overflow or falling edge of ECT4ET pin when E4_EXEN2=1.

23.5.1 Capture Mode

In the capture mode there are two options that are selected by bit E4_EXEN2 in T2CON4. If E4_EXEN2=0, then ECT4 is a 16-bit timer or counter (as selected by C/ECT4T in T2CON4) which, upon overflowing sets bit E4_TF2, the ECT4 overflow bit. This bit can be used to generate an interrupt (by enabling the ECT4 interrupt bit in the EXINTEN register). If E4_EXEN2= 1, ECT4 operates as described above, but with the added feature that a 1- to -0 transition at external input ECT4ET causes the current value in the ECT4 registers, TL2_4(0XDC) and TH2_4(0XDD), to be captured into registers RLDL_4(0XDA) and RLDH_4(0XDB), respectively. In addition, the transition at ECT4ET causes bit E4_EXF2 in T2CON4 to be set, and E4_EXF2 like E4_TF2 can generate an interrupt (which vectors to the same location as ECT4 overflow interrupt.

The ECT4 interrupt service routine can interrogate E4_TF2 and E4_EXF2 to determine which event caused the interrupt). The capture mode is illustrated in Figure (There is no reload value for TL2_4 and TH2_4 in this mode. Even when a capture event occurs from ECT4ET, the counter keeps on counting ECT4ET pin transitions or osc/12 pulses.)

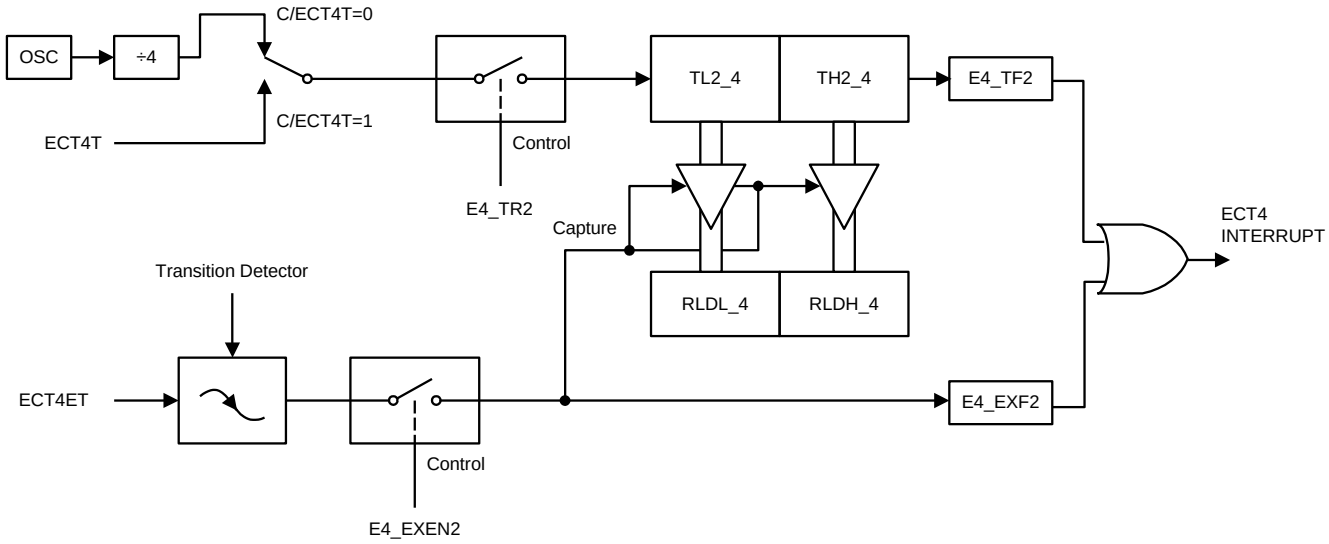


Figure-65. ECT4 Capture Mode

23.5.2 Auto-Reload Mode

In the 16-bit auto-reload mode, ECT4 can be configured (as either a timer or counter (C/ECT4T in T2CON4)). ECT4 will count up automatically. In this mode there are two options selected by bit E4_EXEN2 in T2CON4 register. If E4_EXEN2=0, then ECT4 counts up to 0XFFFF and sets the E4_TF2 (Overflow Flag) bit upon overflow. This causes the ECT4 registers to be reloaded with the 16-bit value in RLDL_4 and RLDH_4. The values in RLDL_4 and RLDH_4 are preset by software means. If E4_EXEN2=1, then a 16-bit reload can be triggered either by an overflow or by a 1-to-0 transition at input ECT4ET. This transition also sets the E4_EXF2 bit. The ECT4 interrupt, if enabled, can be generated when either E4_TF2 or E4_EXF2 are 1.

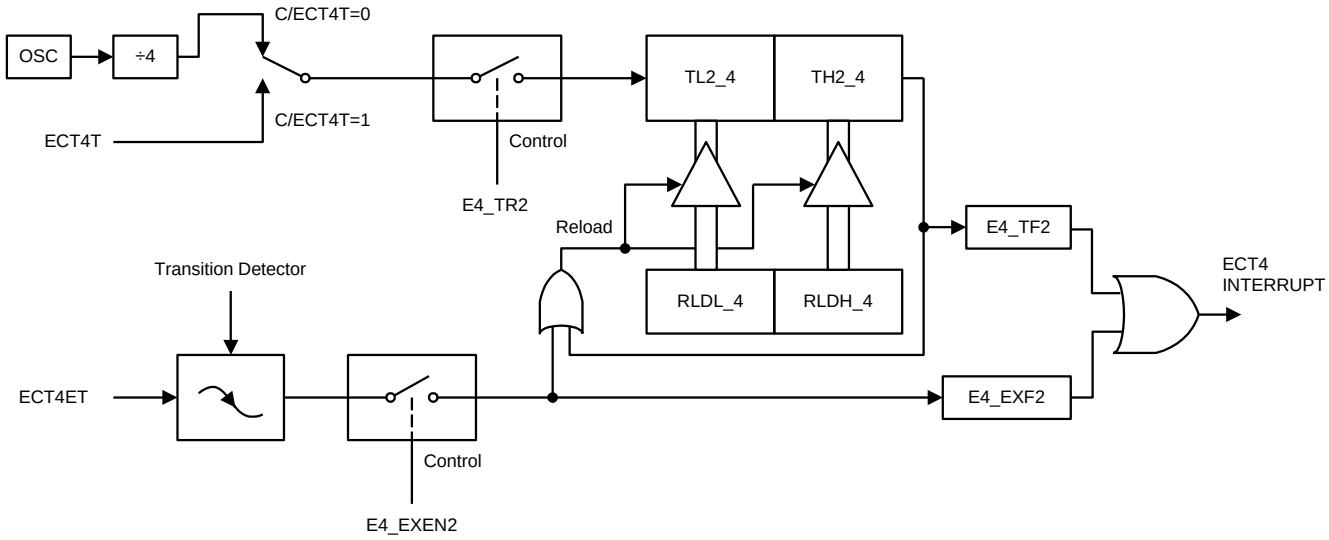


Figure-66. ECT4 Auto-Reloaded Mode

23.6 ECT5

ECT5 is a 16-bit Timer/Counter, which can operate as either an event timer or an event counter, as selected by C/ECT5T* in the special function register T2CON5 (0XE1). ECT5 has two operating modes: Capture, Auto-reload, which are selected by bits in the T2CON5.

T2CON5 (r/w) : (SFR 0XE1) ECT5 Configuration Register (default 0X00)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	E5_TF2	E5_EXF2	E5_TF2EN	E5_EXF2EN	E5_EXEN2	E5_TR2	C/ECT5T	E5_CP/RL2
Default	0	0	0	0	0	0	0	0

E5_TF2 : ECT5 interrupt (overflow) flag. Must be cleared by software.
E5_EXF2 : Falling edge indicator on ECT5ET pin when E5_EXEN2=1. Must be cleared by software.
E5_TF2EN : Overflow interrupt enable.
=1 → Enable overflow interrupt
=0 → Disable overflow interrupt
E5_EXF2EN : Falling edge interrupt enable
=1 → Enable falling edge interrupt
=0 → Disable falling edge interrupt
E5_EXEN2 : Enable ECT5ET pin functionality.
=1 → Allows capture or reload as a result of ECT5ET pin falling edge
=0 → Ignore ECT5ET events
E5_TR2 : Start/stop ECT5
=1 → Start
=0 → Stop
C/ECT5T : Timer/counter select
=1 → External event counter. Clock source is ECT5T pin
=0 → Timer Internally clocked
E5_CP/RL2 : Capture/reload select
=1 → ECT5ET pin falling edge causes capture to occur when E5_EXEN2=1
=0 → Automatic reload occurs: on ECT5 overflow or falling edge of ECT5ET pin when E5_EXEN2=1.

23.6.1 Capture Mode

In the capture mode there are two options that are selected by bit E5_EXEN2 in T2CON5. If E5_EXEN2=0, then ECT5 is a 16-bit timer or counter (as selected by C/ECT5T in T2CON5) which, upon overflowing sets bit E5_TF2, the ECT5 overflow bit. This bit can be used to generate an interrupt (by enabling the ECT5 interrupt bit in the EXINTEN register). If E5_EXEN2= 1, ECT5 operates as described above, but with the added feature that a 1- to -0 transition at external input ECT5ET causes the current value in the ECT5 registers, TL2_5(0XE4) and TH2_5(0XE5), to be captured into registers RLDL_5(0XE2) and RLDH_5(0XE3), respectively. In addition, the transition at ECT5ET causes bit E5_EXF2 in T2CON5 to be set, and E5_EXF2 like E5_TF2 can generate an interrupt (which vectors to the same location as ECT5 overflow interrupt.

The ECT5 interrupt service routine can interrogate E5_TF2 and E5_EXF2 to determine which event caused the interrupt). The capture mode is illustrated in Figure (There is no reload value for TL2_5 and TH2_5 in this mode. Even when a capture event occurs from ECT5ET, the counter keeps on counting ECT5ET pin transitions or osc/12 pulses.).

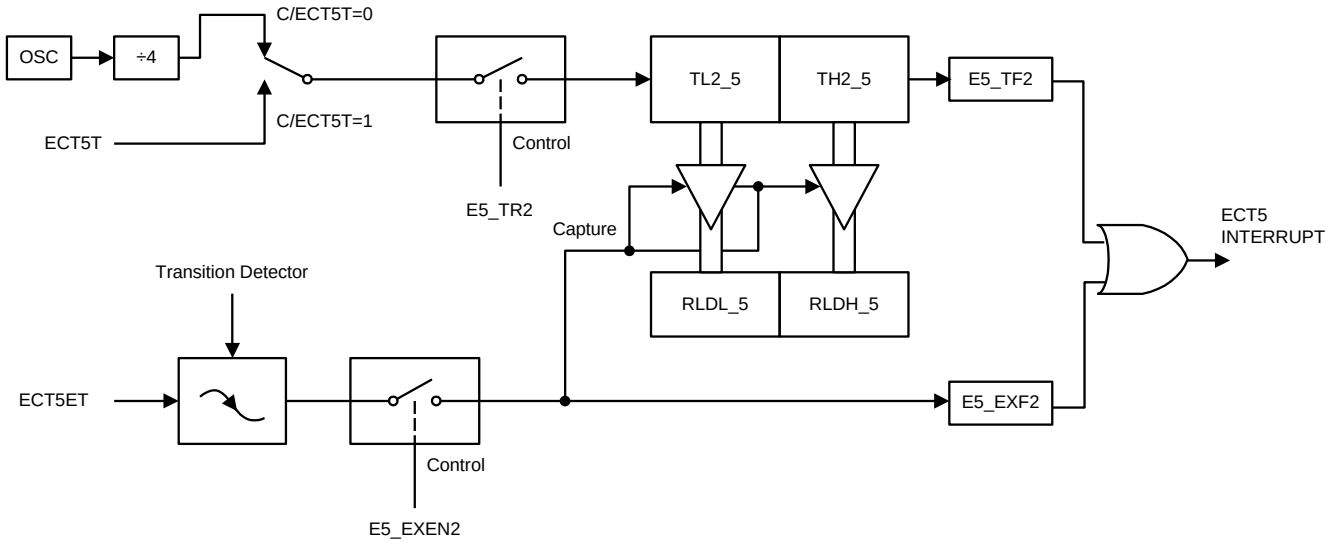


Figure-67. ECT5 Capture Mode

23.6.2 Auto-Reload Mode

In the 16-bit auto-reload mode, ECT5 can be configured (as either a timer or counter (C/ECT5T in T2CON5)). ECT5 will count up automatically. In this mode there are two options selected by bit E5_EXEN2 in T2CON5 register. If E5_EXEN2=0, then ECT5 counts up to 0XFFFF and sets the E5_TF2 (Overflow Flag) bit upon overflow. This causes the ECT5 registers to be reloaded with the 16-bit value in RLDL_5 and RLDH_5. The values in RLDL_5 and RLDH_5 are preset by software means. If E5_EXEN2=1, then a 16-bit reload can be triggered either by an overflow or by a 1-to-0 transition at input ECT5ET. This transition also sets the E5_EXF2 bit. The ECT5 interrupt, if enabled, can be generated when either E5_TF2 or E5_EXF2 are 1.

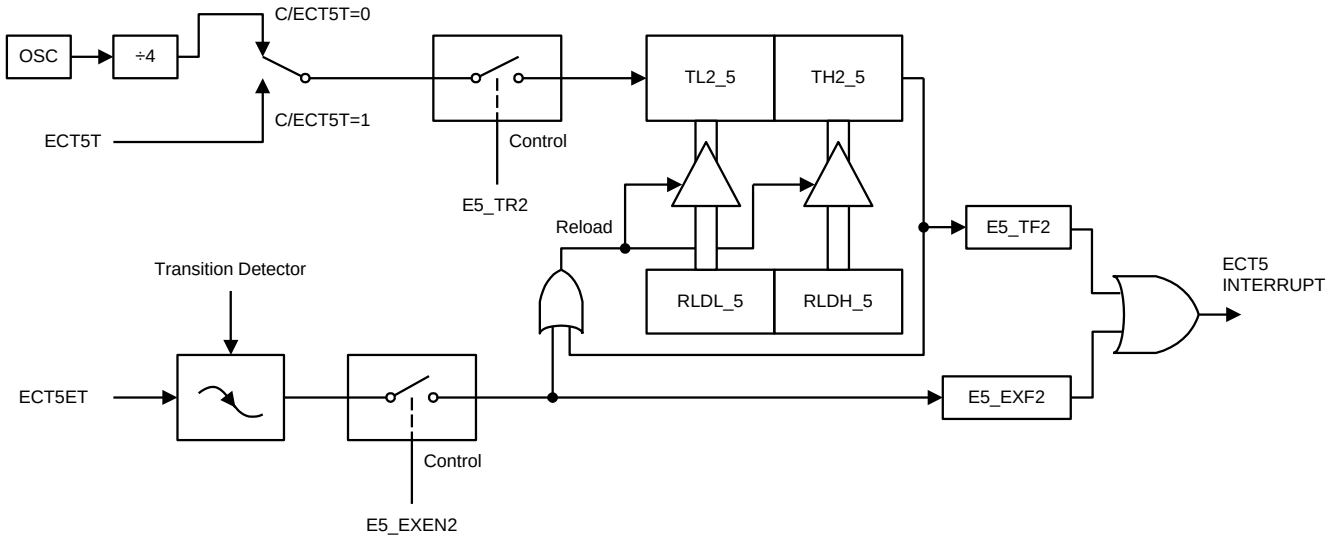
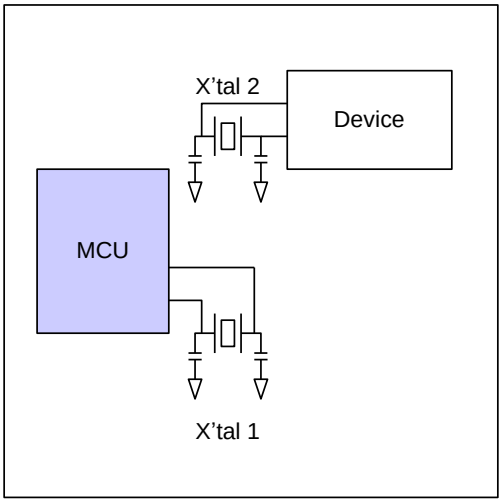


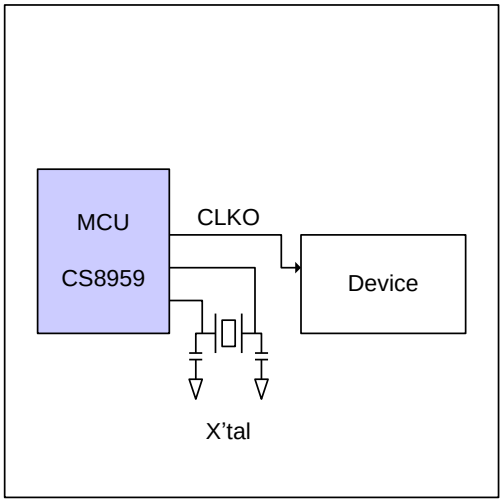
Figure-68. ECT5 Auto-Reloaded Mode

24. BUFFERED CLOCK OUTPUT

There is a buffered clock output port can be optionally defined (see PADMOD52, F52h) on pins P8.3 (CLKO) .The output frequencies of CLKO depend on the external crystal/oscillator frequency (Fxtal) on pins X1/X2, and can be further programmed to Fxtal / 1, Fxtal / 2, Fxtal / 4, or Fxtal / 8 by properly settings on register CLKOSEL (F55h). In application these clock output ports can be used as clock sources of peripheral devices. Users therefore can save the crystal/oscillator amount on boards. The following blocks illustrate the connection in application.



Prior Solution: 2 X'tal required at least



CS8959 Solution: 1 X'tal required only

Figure-69. CLKO Configuration

SEL (r/w) : (XFR 0X0F55) RTC and CLKO select (default 0X20)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	-	TRTC	WKOSC	LPS	CKSL1	CKSL2	-	-
Default	-	0	1	0	0	0	-	-

TRTC = 1 → RTC clock source from internal clock generator
= 0 → RTC clock source from RTC_X1
WKOSC Write 1 to wake up the external clock X1 from Low power mode.
LPS Low power mode select
= 0 → The system clock source is from external clock input X1
= 1 → The system clock source is from RTC_X1 / 2
CKSL2, CKSL1 = 00 → CLKO is system clock
= 01 → CLKO is system clock divided 4
= 10 → CLKO is system clock divided 2
= 11 → CLKO is system clock divided 8

CLKO Frequencies

CKSL2 (SEL bit-2)	CKSL1 (SEL bit-3)	CLKO frequencies
0	0	CLK
0	1	CLK/4
1	0	CLK/2
1	1	CLK/8

25. JTAG TAP CONTROLLER

CS8959 provides the JTAG interface with the TAP controller compliant to IEEE 1149.1, with which the following functions are implemented.

- Boundary Scan: The value of the IO pins could be read or written via the JTAG boundary-scan chain, as the MCU remains in its functional mode.
- Debug: With the software debug tool, user could run or stop the MCU, set breakpoint, read or write the register, and go or step his program.
- MBIST: The JTAG TAP controller allows user access to the MBIST (Memory built-in self test) scan chain to test the SRAM memory.
- ISP: The user could easily download his program to the MCU via the JTAG interface with the software ISP tool.

If you need any tool of above, you can ask ISSI f*or help.*

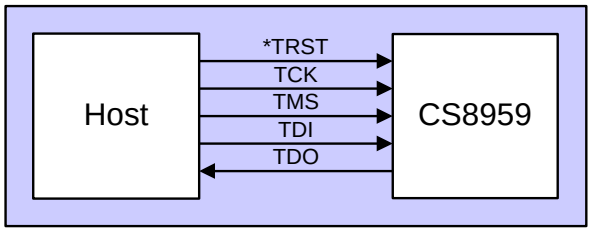


Figure-70. Connection via the JTAG interface

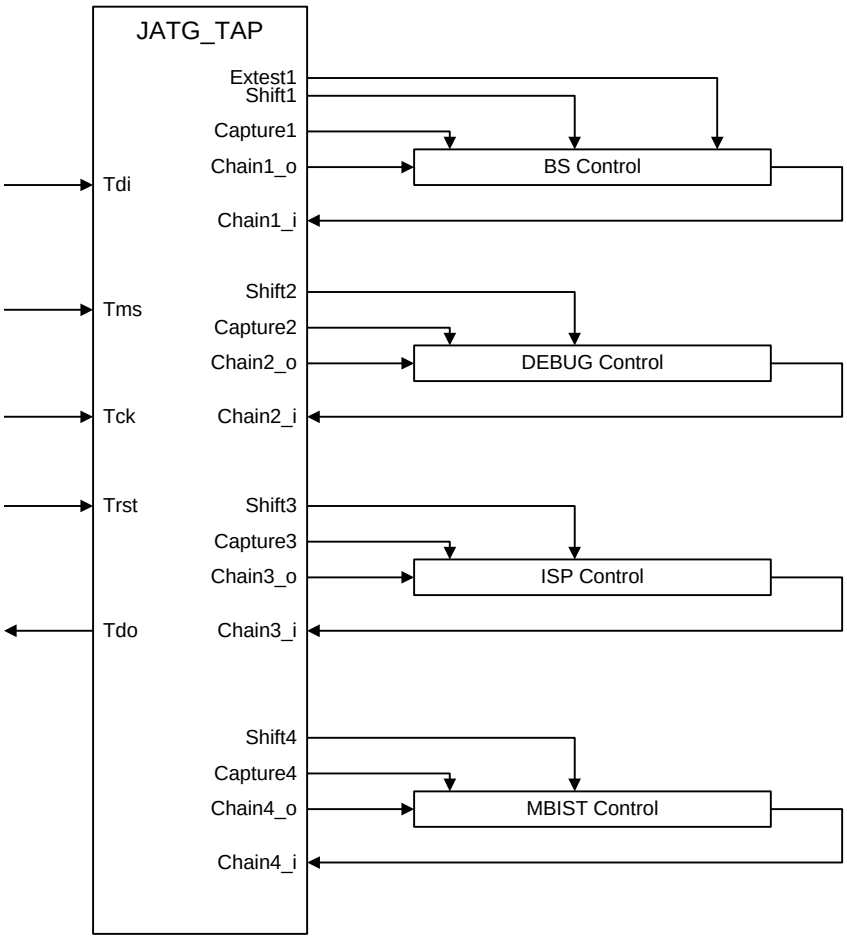


Figure-71. Diagram of the JTAG architecture

JTAG interface signals

Pin Name	Abbreviation	Signal	Description
JTAG_CLK	TCK	Test Clock	A Test clock input that synchronizes the test logic.
JTAG_SEL	TMS	Test Mode State	A pulled-up input that is sampled at the rising edge of TCK to determine the next state.
JTAG_IN	TDI	Test Data In	A pulled-up Test Data input that is sampled at the rising edge of TCK.
JTAG_OUT	TDO	Test Data Out	Represents the Test Data shifted out of the device and is valid on the falling edge of TCK.
JTAG_RST	TRST	Test Reset	An optional pin* which, when available, can reset (asynchronous, active high) the TAP controller's state machine.

**If JTAG_RST pin is not presented on the host device, the TAP controller could also be reset by pulling TMS for 5 TCK cycles, which avoids resetting the whole chip.*

26. CAN CONTROLLER

The Controller Area Network (CAN) is a serial, asynchronous, multi-master communication protocol for connecting electronic control modules, sensors and actuators in automotive and industrial applications.

With the CAN gateway, we provide two CAN controllers which can connected two different networks. The two controller is designed according to CAN protocol version 2.0B.
Attractive features are implemented for a wide range of applications, supporting system optimization, diagnosis and maintenance.

26.1 CAN Controller Function Description

With the CS8959, we provide two CAN controllers, CAN0 and CAN1 for short, which act as gateway, connecting two different networks. They also can play the stand-alone node role.

When acting as nodes, they are dependent on each other and completely have the same function. So, if there are no special notes in this specification, the function and registers descriptions apply to the two CAN controllers.
In CS8959, each CAN controller has the main function as follows:

- Operating mode
- Silent mode
- Listen-Only mode
- Bus Auto-Recovery
- Self test mode
- Self-Reception
- Programmable Baud-Rate
- 32 filters
- Dispatch
- Forward
- Error Detection
- Loop test
- Back-up pins

26.2 Registers Description

26.2.1 Registers mapping

The CAN0 and CAN1 are the peripherals of the MCU. CAN0 occupies space from 0X0B00 to 0X0CFF in XFR (Auxiliary Function Register). CAN1 occupies space from 0X0D00 to 0X0EFF. Following tables list these registers. The address reserved is for future use. Reading and writing operation to those addresses has no effect.

Note that some registers are accessible in operating mode and some registers are accessible in Silent mode. The sign "—" stands for inaccessible in writing operation.

CAN0 Registers Mapping

ADDR	Operating Mode		Silent Mode	
	read	write	read	write
0B00H	Mode Register (Note 1)	Mode Register (Note 1)	Mode Register (Note 1)	Mode Register (Note 1)
0B01H	00H (Note 2)	Command Register (Note 2)	00H (Note 2)	——
0B02H	Status Register	——	Status Register	——
0B03H	Interrupt Register (Note 3)	——	00H (Note 3)	——
0B04H	Interrupt Enable Register	Interrupt Enable Register	Interrupt Enable Register	Interrupt Enable Register
0B05H	Forward Enable Register 3	——	Forward Enable Register 3	Forward Enable Register 3
0B06H	Forward Enable Register 2	——	Forward Enable Register 2	Forward Enable Register 2
0B07H	Forward Enable Register 1	——	Forward Enable Register 1	Forward Enable Register 1
0B08H	Forward Enable Register 0	——	Forward Enable Register 0	Forward Enable Register 0
0B09H	Bus Timing Register 0	——	Bus Timing Register 0	Bus Timing Register 0
0B0AH	Bus Timing Register 1	——	Bus Timing Register 1	Bus Timing Register 1
0B0BH	Reserved	Reserved	Reserved	Reserved
0B0CH	Filer Enable Register 3	——	Filer Enable Register 3	Filer Enable Register 3
0B0DH	Filer Enable Register 2	——	Filer Enable Register 2	Filer Enable Register 2
0B0EH	Filer Enable Register 1	——	Filer Enable Register 1	Filer Enable Register 1
0B0FH	Filer Enable Register 0	——	Filer Enable Register 0	Filer Enable Register 0
0B10H	Arbitration Lost Capture Register	——	Arbitration Lost Capture Register	——
0B11H	Error Code Capture Register	——	Error Code Capture Register	——
0B12H	Error Warning Limit Register	——	Error Warning Limit Register	Error Warning Limit Register
0B13H	Receive Error Counter Register	——	Receive Error Counter Register	Receive Error Counter Register
0B14H	Transmit Error Counter Register	——	Transmit Error Counter Register	Transmit Error Counter Register
0B15H	Receive Frame Counter Register	——	00H(Note 4)	00H
0B16H	Filter Number Register	——	Filter Number Register	——
0B17H	Receive Frame Information Register	Transmit Frame Information Register	Receive Frame Information Register	——
0B18H	Receive ID Register 3	Transmit ID Register 3	Receive ID Register 3	——
0B19H	Receive ID Register 2	Transmit ID Register 2	Receive ID Register 2	——
0B1AH	Receive ID Register 1	Transmit ID Register 1	Receive ID Register 1	——

ADDR	Operating Mode		Silent Mode	
	read	write	read	write
0B1BH	Receive ID Register 0	Transmit ID Register 0	Receive ID Register 0	___
0B1CH	Receive DATA Register 1	Transmit DATA Register 1	Receive DATA Register 1	___
0B1DH	Receive DATA Register 2	Transmit DATA Register 2	Receive DATA Register 2	___
0B1EH	Receive DATA Register 3	Transmit DATA Register 3	Receive DATA Register 3	___
0B1FH	Receive DATA Register 4	Transmit DATA Register 4	Receive DATA Register 4	___
0B20H	Receive DATA Register 5	Transmit DATA Register 5	Receive DATA Register 5	___
0B21H	Receive DATA Register 6	Transmit DATA Register 6	Receive DATA Register 6	___
0B22H	Receive DATA Register 7	Transmit DATA Register 7	Receive DATA Register 7	___
0B23H	Receive DATA Register 8	Transmit DATA Register 8	Receive DATA Register 8	___
0B24H	Filter Status Register 3	___	Dispatch Time Register 3(Note5)	Dispatch Time Register 3
0B25H	Filter Status Register 2	___	Dispatch Time Register 2	Dispatch Time Register 2
0B26H	Filter Status Register 1	___	Dispatch Time Register 1	Dispatch Time Register 1
0B27H	Filter Status Register 0	___	Dispatch Time Register 0	Dispatch Time Register 0
0B28H | 0C27H	Receive FIFO	Receive FIFO	Filter Registers	Filter Registers
0C28H | 0C90H	Reserved			
0C91H	Loop Test Enable	Loop Test Enable	Loop Test Enable	Loop Test Enable
0C92H	Loop Test	Loop Test	Loop Test	Loop Test
0C93H | 0CFFH	Reserved			

Notes:

1. *Bits from Mode[1] to Mode[3] can be read and written in Silent mode and are read only in Operating mode.*
2. *Command Register can only be written in Operating mode. Reading the register in Operating mode will get 0. In Silent mode, the register is cleared.*
3. *This is a read only register. In Silent mode, the register is cleared.*
4. *This is a read only register. In Silent mode, the register is cleared.*
5. *In Silent mode, the upper 4 bits of the Dispatch Time 3 register are safe bits. If you want to configure the Dispatch Time registers, you must write 6h into the upper 4 bits first, or writing operation to the dispatch time registers is invalid.*

CAN 1 Registers Mapping

ADDR	Operating Mode		Silent Mode	
	read	write	read	write
0D00H	Mode Register (Note 1)	Mode Register (Note 1)	Mode Register (Note 1)	Mode Register (Note 1)
0D01H	00H (Note 2)	Command Register (Note 2)	00H (Note 2)	——
0D02H	Status Register	——	Status Register	——
0D03H	Interrupt Register (Note 3)	——	00H (Note 3)	——
0D04H	Interrupt Enable Register	Interrupt Enable Register	Interrupt Enable Register	Interrupt Enable Register
0D05H	Forward Enable Register 3	——	Forward Enable Register 3	Forward Enable Register 3
0D06H	Forward Enable Register 2	——	Forward Enable Register 2	Forward Enable Register 2
0D07H	Forward Enable Register 1	——	Forward Enable Register 1	Forward Enable Register 1
0D08H	Forward Enable Register 0	——	Forward Enable Register 0	Forward Enable Register 0
0D09H	Bus Timing Register 0	——	Bus Timing Register 0	Bus Timing Register 0
0D0AH	Bus Timing Register 1	——	Bus Timing Register 1	Bus Timing Register 1
0D0BH	Reserved	Reserved	Reserved	Reserved
0D0CH	Filer Enable Register 3	——	Filer Enable Register 3	Filer Enable Register 3
0D0DH	Filer Enable Register 2	——	Filer Enable Register 2	Filer Enable Register 2
0D0EH	Filer Enable Register 1	——	Filer Enable Register 1	Filer Enable Register 1
0D0FH	Filer Enable Register 0	——	Filer Enable Register 0	Filer Enable Register 0
0D10H	Arbitration Lost Capture Register	——	Arbitration Lost Capture Register	——
0D11H	Error Code Capture Register	——	Error Code Capture Register	——
0D12H	Error Warning Limit Register	——	Error Warning Limit Register	Error Warning Limit Register
0D13H	Receive Error Counter Register	——	Receive Error Counter Register	Receive Error Counter Register
0D14H	Transmit Error Counter Register	——	Transmit Error Counter Register	Transmit Error Counter Register
0D15H	Receive Frame Counter Register	——	00H(Note 4)	00H
0D16H	Filter Number Register	——	Filter Number Register	——
0D17H	Receive Frame Information Register	Transmit Frame Information Register	Receive Frame Information Register	——
0D18H	Receive ID Register 3	Transmit ID Register 3	Receive ID Register 3	——
0D19H	Receive ID Register 2	Transmit ID Register 2	Receive ID Register 2	——
0D1AH	Receive ID Register 1	Transmit ID Register 1	Receive ID Register 1	——

ADDR	Operating Mode		Silent Mode	
	read	write	read	write
0D1BH	Receive ID Register 0	Transmit ID Register 0	Receive ID Register 0	___
0D1CH	Receive DATA Register 1	Transmit DATA Register 1	Receive DATA Register 1	___
0D1DH	Receive DATA Register 2	Transmit DATA Register 2	Receive DATA Register 2	___
0D1EH	Receive DATA Register 3	Transmit DATA Register 3	Receive DATA Register 3	___
0D1FH	Receive DATA Register 4	Transmit DATA Register 4	Receive DATA Register 4	___
0D20H	Receive DATA Register 5	Transmit DATA Register 5	Receive DATA Register 5	___
0D21H	Receive DATA Register 6	Transmit DATA Register 6	Receive DATA Register 6	___
0D22H	Receive DATA Register 7	Transmit DATA Register 7	Receive DATA Register 7	___
0D23H	Receive DATA Register 8	Transmit DATA Register 8	Receive DATA Register 8	___
0D24H	Filter Status Register 3	___	Dispatch Time Register 3(Note5)	Dispatch Time Register 3
0D25H	Filter Status Register 2	___	Dispatch Time Register 2	Dispatch Time Register 2
0D26H	Filter Status Register 1	___	Dispatch Time Register 1	Dispatch Time Register 1
0D27H	Filter Status Register 0	___	Dispatch Time Register 0	Dispatch Time Register 0
0D28H | 0E27H	Receive FIFO	Receive FIFO	Filter Registers	Filter Registers
0E28H | 0EFFH	Reserved			

Notes:

1. *Bits from Mode[1] to Mode[3] can be read and written in Silent mode and are read only in Operating mode.*
2. *Command Register can only be written in Operating mode. Reading the register in Operating mode will get 0. In Silent mode, the register is cleared.*
3. *This is a read only register. In Silent mode, the register is cleared.*
4. *This is a read only register. In Silent mode, the register is cleared.*
5. *In Silent mode, the upper 4 bits of the Dispatch Time 3 register are safe bits. If you want to configure the Dispatch Time registers, you must write 6h into the upper 4 bits first, or writing operation to the dispatch time registers is invalid.*

26.2.2 Registers Function Description

We take CAN0 as an example in the “Register Function Description” section. If there is no special note, the registers function description also applies to CAN 1.

Mode Register

Mode Register (MOD) XFR ADDR: 0B00H

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
MOD.7		Reserved	0		Reserved
MOD.6		Reserved	0		Reserved
MOD.5		Reserved	0		Reserved
MOD.4		Reserved	0		Reserved
MOD.3	BAR	Bus Auto Recovery	0	1	Bus Auto Recovery Mode (Note 1)
				0	Normal
MOD.2	STM	Self Test Mode	0	1	Self Test Mode (Note 2)
				0	Normal
MOD.1	LOM	Listen Only Mode	0	1	Listen Only Mode (Note 3)
				0	Normal
MOD.0	SM	Silent Mode	1	1	Silent Mode (Note 4)
				0	Operating Mode

Notes:

1. *The bit can be read and written in Silent mode and is read-only in Operating mode. When this bit is set, CAN controller will work in Bus Auto Recovery mode. When bus is off and CAN controller works in this mode, the controller will wait 128 occurrences of bus-free signal (11 consecutive ‘recessive’ bits) and then the bus is turned on again, the value of the Receive Error counter register and Transmit error counter register is 0 and 128 correspondingly. When the bus is off and the CAN controller is not in Bus Auto Recovery mode, the controller enters into Silent mode and the value of the Receive Error counter register and Transmit error counter register is 0 and 128 correspondingly. In this condition, MCU can write a value less then 255 into the Transmit Error Counter register to make the bus on. CAN controller will stay in Silent mode until the CPU clears the Silent mode bit.*
2. *The bit can be read and written in Silent mode and is read only in Operating mode. When this bit is set, CAN controller will work in Self Test Mode. In this mode, a full node test is possible without any other active node on the bus using the self reception request command; The CAN controller will perform a successful transmission, even if there is no acknowledge received. But when transmitting successfully, the transmit error counter will count down by 1 if its value is not zero.*
3. *The bit can be read and written in Silent mode and is read-only in Operating mode. When this bit is set, CAN controller will work in Listen Only mode. In this mode, the CAN controller would give no acknowledge to the CAN-bus, even if a message is received successfully. In this mode, the error counters are stopped at the current value and message transmission is not possible. But when detecting bus error, node will transmit error flag. All other functions can be used like in Operating mode.*
4. *When this bit is set, CAN controller will work in Silent mode. After hardware reset, the CAN controller works in Silent mode. In Silent mode, normal transmitting and receiving is impossible. After CAN controller quit Silent mode, a bus-free signal (11 recessive bits) have to be detected for bus idle.*

Command Register

Command Register (CMD) XFR ADDR: 0B01H

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
CMD.7		Reserved	0	0	Reserved
CMD.6	FR	Forward Request	0	1	Request forwarding a frame (Note 1)
				0	No action
CMD.5	OFR	Overload Frame Request	0	1	Request transmitting overload frame (Note 2)
				0	No action
CMD.4	SRR	Self Receive Request	0	1	A message shall be transmitted and received simultaneously (Note 3)
				0	No action
CMD.3	CDO	Clear Data Overrun	0	1	Clear data overrun status bit (Note 4)
				0	No action
CMD.2	RRB	Release Receive Buffer	0	1	Release receive buffer (Note 5)
				0	No action
CMD.1	AT	Abort Transmitting	0	1	Abort transmitting (Note 6)
				0	No action
CMD.0	TR	Transmit Request	0	1	Transmit request (Note 7)
				0	No action

Notes:

1. *If a frame is received by a filter in CAN0 (CAN1), and the Forward Enable bit of a filter is set, the Forward flag of the frame will be set. When the Forward request bit is set, the frame will be transmitted by CAN1 (CAN0). So a frame will not be forwarded unless the Forward flag and Forward request bit are 1.*
2. *When the bit is set, the start of an Overload frame is only allowed to be started at the first bit time of an expected INTERMISSION.*
3. *Upon self reception request a message is transmitted and simultaneously received if the acceptance filter is set to the corresponding identifier. A receive and a transmit interrupt will indicate correct self reception (see also Self Test Mode in MODE register).*
4. *This command bit is used to clear the data overrun condition indicated by the data overrun status bit.*
5. *After reading the contents of the receive buffer, the CPU can release this memory space in the RXFIFO by setting the Release Receive Buffer bit to logic 1.This may result in another message becoming immediately available within the receive buffer. If there is no other message available, the receive interrupt bit is reset.*
6. *The abort transmission bit is used when the CPU requires the suspension of the previously requested transmission, e.g. to transmit a more urgent message before. A transmission already in progress is not stopped. In order to see if the original message has been either transmitted successfully or aborted, the transmission complete status bit should be checked. This should be done after the transmit buffer status bit has been set to logic 1 or a transmit interrupt has been generated.*
7. *We can transmit a message in the transmit buffer by setting this bit to logic 1. If the transmission request was set to logic 1 in the previous command, it should be cancelled by setting the transmission request bit to logic 0. The requested transmission may be cancelled by setting the abort transmission bit to logic 1.*

Status Register

Status Register (STR) XFR ADDR: 0B02H

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
STA.7	BS	Bus Status (Note 1)	0	1	Bus-off: the CAN controller is not involved in bus activities.
				0	Bus-on: the CAN controller is involved in bus activities.
STA.6	ES	Error Status (Note 2)	0	1	Error: at least one of the error counters has reached or exceeded the CPU warning limit defined by the Error Warning Limit Register
				0	Ok: both error counters are below the warning limit.
STA.5	TS	Transmit Status (Note 3)	0	1	Transmit: the CAN controller is transmitting a message
				0	Idle
STA.4	RS	Receive Status (Note 3)	0	1	Receive: the CAN controller is receiving a message.
				0	Idle
STA.3	TCS	Transmit Complete Status (Note 4)	1	1	Complete: last requested transmission has been successfully completely
				0	Incomplete: previously requested transmission is not yet completed
STA.2	TBS	Transmit Buffer Status (Note 5)	1	1	Released: the CPU may write a message into the transmit buffer
				0	Locked: the CPU cannot access the transmit buffer; a message is either waiting for transmission or in the progress of being transmitted.
STA.1	DOS	Data Overrun Status (Note 6)	0	1	Overrun: a message was lost because there was not enough space for that message in the RXFIFO
				0	Full: one or more complete messages are available in the RXFIFO
STA.0	RBS	Receive Buffer Status (Note 7)	0	1	Full: one or more complete message are available in the RXFIFO
				0	Empty: no message is available

Notes:

1. *When the transmit error counter exceeds the limit of 255, the bus status bit is set to logic 1 CAN controller will wait the minimum protocol-defined time (128 occurrences of the bus-free signal) for bus on again or when controller not in Bus Auto Recovery, CPU can write a value less than 255 into the transmit error counter to make the bus on. Then the bus status bit is cleared (bus-on), the error status bit is set to logic 0 (ok), an error warning interrupt is generated, if enabled.*
2. *Errors detected during reception or transmission will affect the error counters according to the CAN 2.0B protocol specification. The error status bit is set when at least one of the error counters has reached or exceeded the CPU warning limit (EWLR). An error warning interrupt is generated, if enabled. The default value of EWLR after hardware reset is 96.*
3. *If both the receive status and the transmit status bits are logic 0 (idle) the CAN-bus is idle. If at least one bit is set, the controller is waiting to become idle again.*
4. *The transmission complete status bit is set to logic 0 (incomplete) whenever the transmission request bit or the self reception request bit is set to logic 1. The transmission complete status bit will remain at logic 0 until a message is transmitted successfully.*
5. *If the CPU tries to write to the transmit buffer when the transmit buffer status bit is logic 0 (locked), the written byte will not be accepted and will be lost without this being indicated.*
6. *When a message that is to be received has passed the acceptance filter successfully, the CAN controller needs space in the RXFIFO to store the message information and for each data byte, which has been received. If there is not enough space to store the message, that message is dropped and the data overrun condition is indicated to the CPU at the moment this message becomes invalid. If this message is not completed successfully (e.g. due to an error), no overrun condition is indicated.*
7. *After reading all messages within the RXFIFO and releasing their memory space with the command release receive buffer this bit is cleared.*

Interrupt Register

Interrupt Register (IR) XFR ADDR: 0B03H

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
IR.7	BEI	Bus Error Interrupt (Note 1)	0	1	Set: this bit is set when the CAN controller detects an error on the CAN-bus and the BEIE bit is set within the interrupt enable register
				0	Reset
IR.6	ALI	Arbitration Lost Interrupt (Note 1)	0	1	Set: this bit is set when the CAN controller lost the arbitration and become a receiver and the ALIE bit is set to logic 1 within the Interrupt Enable register.
				0	Reset
IR.5	EPI	Error Passive Interrupt (Note 1)	0	1	Set: this bit is set whenever the CAN controller has reached the error passive status (at least one error counter exceeds the protocol-defined level of 127) or if the CAN controller is in the error passive status and enters the error active status again and the EPIE bit is set within the interrupt enable register.
				0	Reset
IR.4	Reserved	Reserved	0	0	Reserved
IR.3	DOI	Data Overrun Interrupt (Note 1)	0	1	Set: this bit is set on a '0-to-1' transition of the data overrun status bit and the DOIE bit is set within the interrupt enable register
				0	Reset
IR.2	EI	Error Warning Interrupt (Note 1)	0	1	Set: this bit is set on every change (set and clear) of either the error status or bus status bits and the EIE bit is set within the interrupt enable register
				0	Reset
IR.1	TI	Transmit Interrupt (Note 1)	0	1	Set: this bit is set whenever the transmit buffer status changes from '0-to-1' (released) and the TIE bit is set within the interrupt enable register.
				0	Reset
IR.0	RI	Receive Interrupt (Note 1,2)	0	1	Set: this bit is set while the receive FIFO is not empty and the RIE bit is set within the interrupt enable register.
				0	Reset

Notes:

1. *The interrupt register allows the identification of an interrupt source. When one or more bits of this register are set to logic 1, a CAN interrupt will be indicated to the CPU. After CPU reads this register, all bits are cleared except for the Receive Interrupt bit. The interrupt register appears to the CPU as a read only memory.*
2. *The behavior of this bit is equivalent to that of the receive buffer status bit if the corresponding interrupt enable bit (RIE) is set. So the receive interrupt bit is not cleared upon a read access to the interrupt register. Giving the command 'Release Receive Buffer' will clear RI temporarily. If there is another message available within the FIFO after the release command, RI is set again. Otherwise RI remains cleared.*

Interrupt Enable Register

Interrupt Register Interrupt Enable Register (IER) XFR ADDR : 0B04H

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
IER.7	BEIE	Bus Error Interrupt Enable (Note 1)	0	1	Enabled: if a bus error has been detected, the CAN controller requests the respective interrupt.
				0	Disabled
IER.6	ALIE	Arbitration Lost Interrupt Enable (Note 1)	0	1	Enabled: if the CAN controller has lost arbitration, the respective interrupt is requested.
				0	Disabled
IER.5	EPIE	Error Passive Interrupt Enable (Note 1)	0	1	Enabled: if the error status of the CAN controller changes from error active to error passive or vice versa, the respective interrupt is requested.
				0	Disabled
IER.4	Reserved	Reserved	0	0	Reserved
IER.3	DOIE	Data Overrun Interrupt Enable (Note 1)	0	1	Enabled: if the data overrun status bit is set (see status register), the CAN controller requests the respective interrupt.
				0	Disabled
IER.2	EIE	Error Warning Interrupt Enable (Note 1)	0	1	Enabled: if the error or bus status changes, the CAN controller requests the respective interrupt.
				0	Disabled
IER.1	TIE	Transmit Interrupt Enable (Note 1)	0	1	Enabled: when a message has been successfully transmitted or the transmit buffer is accessible again (e. g after an abort transmission command), the CAN controller requests the respective interrupt.
				0	Disabled
IER.0	RIE	Receive Interrupt Enable (Note 1)	0	1	Enabled: when the receive buffer status is 'full', the CAN controller requests the respective interrupt.
				0	Disabled

Notes:

1. *The interrupt enable register allows indicating different types of interrupt source. The register appears to the CPU as a read/write memory.*

Forward Enable Register

Forward Enable Register 3 (FWE3) XFR ADDR: 0B05H (Note 1)

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
FWE3.7	FWE31	Filter 31 forward enable	0	1	Filter 31 forward enabled
				0	Filter 31 forward disabled
FWE3.6	FWE30	Filter 30 forward enable	0	1	Filter 30 forward enabled
				0	Filter 30 forward disabled
FWE3.5	FWE29	Filter 29 forward enable	0	1	Filter 29 forward enabled
				0	Filter 29 forward disabled
FWE3.4	FWE28	Filter 28 forward enable	0	1	Filter 28 forward enabled
				0	Filter 28 forward disabled
FWE3.3	FWE27	Filter 27 forward enable	0	1	Filter 27 forward enabled
				0	Filter 27 forward disabled
FWE3.2	FWE26	Filter 26 forward enable	0	1	Filter 26 forward enabled
				0	Filter 26 forward disabled
FWE3.1	FWE25	Filter 25 forward enable	0	1	Filter 25 forward enabled
				0	Filter 25forward disabled
FWE3.0	FWE24	Filter 24 forward enable	0	1	Filter 24 forward enabled
				0	Filter 24 forward disabled

Note:
1. This register is writeable only in Silent mode. In Operating mode, the register is read-only.

Forward Enable Register 2

Forward Enable Register 2 (FWE2) XFR ADDR: 0B06H (Note 1)

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
FWE2.7	FWE23	Filter 23 forward enable	0	1	Filter 23 forward enabled
				0	Filter 23 forward disabled
FWE2.6	FWE22	Filter 22 forward enable	0	1	Filter 22 forward enabled
				0	Filter 22 forward disabled
FWE2.5	FWE21	Filter 21 forward enable	0	1	Filter 21 forward enabled
				0	Filter 21 forward disabled
FWE2.4	FWE20	Filter 20 forward enable	0	1	Filter 20 forward enabled
				0	Filter 20 forward disabled
FWE2.3	FWE19	Filter 19 forward enable	0	1	Filter 19 forward enabled
				0	Filter 19 forward disabled
FWE2.2	FWE18	Filter 18 forward enable	0	1	Filter 18 forward enabled
				0	Filter 18 forward disabled
FWE2.1	FWE17	Filter 17 forward enable	0	1	Filter 17 forward enabled
				0	Filter 17forward disabled
FWE2.0	FWE16	Filter 16 forward enable	0	1	Filter 16 forward enabled
				0	Filter 16 forward disabled

Note:
1. This register is writeable only in Silent mode. In Operating mode, the register is read-only.

Forward Enable Register 1

Forward Enable Register 1 (FWE1) XFR ADDR: 0B07H (Note 1)

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
FWE1.7	FWE15	Filter 15 forward enable	0	1	Filter 15 forward enabled
				0	Filter 15 forward disabled
FWE1.6	FWE14	Filter 14 forward enable	0	1	Filter 14 forward enabled
				0	Filter 14 forward disabled
FWE1.5	FWE13	Filter 13 forward enable	0	1	Filter 13 forward enabled
				0	Filter 13 forward disabled
FWE1.4	FWE12	Filter 12 forward enable	0	1	Filter 12 forward enabled
				0	Filter 12 forward disabled
FWE1.3	FWE11	Filter 11 forward enable	0	1	Filter 11 forward enabled
				0	Filter 11 forward disabled
FWE1.2	FWE10	Filter 10 forward enable	0	1	Filter 10 forward enabled
				0	Filter 10 forward disabled
FWE1.1	FWE9	Filter 09 forward enable	0	1	Filter 09 forward enabled
				0	Filter 09forward disabled
FWE1.0	FWE8	Filter 08 forward enable	0	1	Filter 08 forward enabled
				0	Filter 08 forward disabled

Note:
1. This register is writeable only in Silent mode. In Operating mode, the register is write-only.

Forward Enable Register 0

Forward Enable Register 0 (FWE0) XFR ADDR: 0B08H (Note 1)

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
FWE0.7	FWE7	Filter 07 forward enable	0	1	Filter 07 forward enabled
				0	Filter 07 forward disabled
FWE0.6	FWE6	Filter 06 forward enable	0	1	Filter 06 forward enabled
				0	Filter 06 forward disabled
FWE0.5	FWE5	Filter 05 forward enable	0	1	Filter 05 forward enabled
				0	Filter 05 forward disabled
FWE0.4	FWE4	Filter 04 forward enable	0	1	Filter 04 forward enabled
				0	Filter 04 forward disabled
FEW0.3	FWE3	Filter 03 forward enable	0	1	Filter 03 forward enabled
				0	Filter 03 forward disabled
FWE0.2	FWE2	Filter 02 forward enable	0	1	Filter 02 forward enabled
				0	Filter 02 forward disabled
FWE0.1	FWE1	Filter 01 forward enable	0	1	Filter 01 forward enabled
				0	Filter 01forward disabled
FWE0.0	FWE0	Filter 00 forward enable	0	1	Filter 00 forward enabled
				0	Filter 00 forward disabled

Note:
1. This register is writeable only in Silent mode. In Operating mode, the register is write-only.

Bus Timing Register

BTR0 (r/w) : (XFR 0X0B09) Bus Timing Register 0 (default 0X00)

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
SJW.1 (Note 1)	SJW.0 (Note 1)	BRP.5 (Note 2)	BRP.4 (Note 2)	BRP.3 (Note 2)	BRP.2 (Note 2)	BRP.1 (Note 2)	BRP.0 (Note 2)

Notes:

1. *To compensate for phase shifts between clock oscillators of different bus controllers, any bus controller must re-synchronize on any relevant signal edge of the current transmission. The synchronization jump width defines the maximum number of clock cycles a bit period may be shortened or lengthened by one re-synchronization:*

$$t_{sjw} = 2 \times SJW.1 + SJW.0 + 1$$

2. *The period of the CAN system clock* t_{SCL} *is programmable and determines the individual bit timing. The CAN system clock is calculated using the following equation:*

$$t_{scl} = 2 \times t_{clk} \times (32 \times BRP.5 + 16 \times BRP.4 + 8 \times BRP.3 + 4 \times BRP.2 + 2 \times BRP.1 + BRP.0 + 1)$$ *Where*

t_{clk} *= time period of the XTAL frequency =* $\frac{1}{f_{XTAL}}$

BTR1 (r/w) : (XFR 0X0B0A) Bus Timing Register 1 (default 0X67)

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
SAM (Note 1)	TSEG2.2 (Note 2)	TSEG2.1 (Note 2)	TSEG2.0 (Note 2)	TSEG1.3 (Note 2)	TSEG1.2 (Note 2)	TSEG1.1 (Note 2)	TSEG1.0 (Note 2)

Notes:

1. *When SAM bit is logic 1, the bus is triple-sampled: the bus is sampled three times, recommended for low/medium speed bus (class A and class B) where it is beneficial for filtering spikes on the bus line. When SAM bit is logic 0, the bus is single-sampled: the bus is sampled once, recommended for high-speed buses (SAE class C).*
2. *TSEG1 and TSEG2 determine the number of clock cycles per bit period and the location of the sample point, where:*

$$t_{SYNCSEG} = 1 \times t_{scl}$$

$$t_{SEG1} = t_{scl} \times (8 \times TSEG1.3 + 4 \times TSEG1.2 + 2 \times TSEG1.1 + TSEG1.0 + 1)$$

$$t_{SEG2} = t_{scl} \times (4 \times TSEG2.2 + 2 \times TSEG2.1 + TSEG2.0 + 1)$$

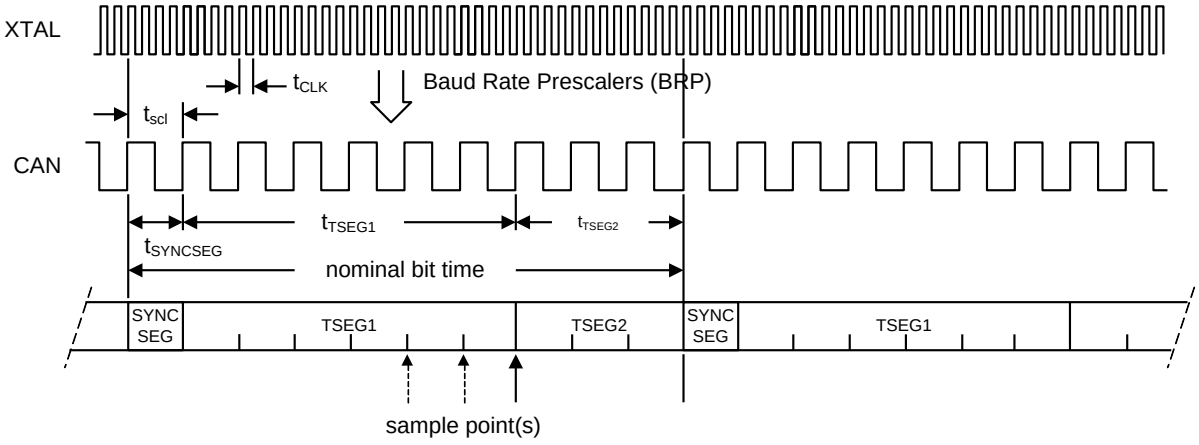


Figure-72. General Structure Of A Bit Period

Filter Enable Register

Filter Enable Register 3 (FTE3) **XFR ADDR: 0B0CH (Note 1)**

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
FTE3.7	FTE31	Filter 31 enable	1	1	Filter 31 enabled
				0	Filter 31 disabled
FTE3.6	FTE30	Filter 30 enable	1	1	Filter 30 enabled
				0	Filter 30 disabled
FTE3.5	FTE29	Filter 29 enable	1	1	Filter 29 enabled
				0	Filter 29 disabled
FTE3.4	FTE28	Filter 28 enable	1	1	Filter 28 enabled
				0	Filter 28 disabled
FTE3.3	FTE27	Filter 27 enable	1	1	Filter 27 enabled
				0	Filter 27 disabled
FTE3.2	FTE26	Filter 26 enable	1	1	Filter 26 enabled
				0	Filter 26 disabled
FTE3.1	FTE25	Filter 25 enable	1	1	Filter 25 enabled
				0	Filter 25 disabled
FTE3.0	FTE24	Filter 24 enable	1	1	Filter 24 enabled
				0	Filter 24 disabled

Note:
1. *This register is write able only in Silent mode. In Operating mode, the register is write-only.*

Filter Enable Register 2 (FTE2) **XFR ADDR: 0B0DH (Note 1)**

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
FTE2.7	FTE23	Filter 23 enable	1	1	Filter 23 enabled
				0	Filter 23 disabled
FTE2.6	FTE22	Filter 22 enable	1	1	Filter 22 enabled
				0	Filter 22 disabled
FTE2.5	FTE21	Filter 21 enable	1	1	Filter 21 enabled
				0	Filter 21 disabled
FTE2.4	FTE20	Filter 20 enable	1	1	Filter 20 enabled
				0	Filter 20 disabled
FTE2.3	FTE19	Filter 19 enable	1	1	Filter 19 enabled
				0	Filter 19 disabled

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
FTE2.2	FTE18	Filter 18 enable	1	1	Filter 18 enabled
				0	Filter 18 disabled
FTE2.1	FTE17	Filter 17 enable	1	1	Filter 17 enabled
				0	Filter 17 disabled
FTE2.0	FTE16	Filter 16 enable	1	1	Filter 16 enabled
				0	Filter 16 disabled

Note:
1. This register is write able only in Silent mode. In Operating mode, the register is write-only.

Filter Enable Register 1 (FTE1) **XFR ADDR: 0B0EH (Note 1)**

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
FTE1.7	FTE15	Filter 15 enable	1	1	Filter 15 enabled
				0	Filter 15 disabled
FTE1.6	FTE14	Filter 14 enable	1	1	Filter 14 enabled
				0	Filter 14 disabled
FTE1.5	FTE13	Filter 13 enable	1	1	Filter 13 enabled
				0	Filter 13 disabled
FTE1.4	FTE12	Filter 12 enable	1	1	Filter 12 enabled
				0	Filter 12 disabled
FTE1.3	FTE11	Filter 11 enable	1	1	Filter 11 enabled
				0	Filter 11 disabled
FTE1.2	FTE10	Filter 10 enable	1	1	Filter 10 enabled
				0	Filter 10 disabled
FTE1.1	FTE9	Filter 09 enable	1	1	Filter 09 enabled
				0	Filter 09 disabled
FTE1.0	FTE8	Filter 08 enable	1	1	Filter 08 enabled
				0	Filter 08 disabled

Note:
1. This register is write able only in Silent mode. In Operating mode, the register is write-only.

Filter Enable Register 0 (FTE0) **XFR ADDR: 0B0FH (Note 1)**

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
FTE0.7	FTE7	Filter 07 enable	1	1	Filter 07 enabled
				0	Filter 07 disabled
FTE0.6	FTE6	Filter 06 enable	1	1	Filter 06 enabled
				0	Filter 06 disabled
FTE0.5	FTE5	Filter 05 enable	1	1	Filter 05 enabled
				0	Filter 05 disabled
FTE0.4	FTE4	Filter 04 enable	1	1	Filter 04 enabled
				0	Filter 04 disabled
FTE0.3	FTE3	Filter 03 enable	1	1	Filter 03 enabled
				0	Filter 03 disabled
FTE0.2	FTE2	Filter 02 enable	1	1	Filter 02 enabled
				0	Filter 02 disabled
FTE0.1	FTE1	Filter 01 enable	1	1	Filter 01 enabled
				0	Filter 01 disabled
FTE0.0	FTE0	Filter 00 enable	1	1	Filter 00 enabled
				0	Filter 00 disabled

Note:
1. This register is write able only in Silent mode. In Operating mode, the register is write-only

Arbitration Lost Capture Register

Arbitration Lost Capture Register (ALC) XFR ADDR: 0B10H

BIT	SYMBOL	NAME	VALUE	FUNCTION
ALC.7~ALC.5	-	Reserved	For value and function see “Function of bits 4 to 0 of the Arbitration Lost Capture register” Table	
ALC.4	BITNO4	Bit number 4		
ALC.3	BITNO3	Bit number 3		
ALC.2	BITNO2	Bit number 2		
ALC.1	BITNO1	Bit number 1		
ALC.0	BITNO0	Bit number 0		

This register contains information about the bit position of losing arbitration. The Arbitration Lost Capture register appears to the CPU as a read only memory. Reserved bits are read as logic 0.

On arbitration lost, the corresponding arbitration lost interrupt is forced, if enabled. At the same time, the current bit position of the bit stream processor is captured into the Arbitration Lost Capture register. The content within this register is fixed until the users software has read out its contents once. The capture mechanism is then activated again.

The corresponding interrupt flag located in the interrupt register is cleared during the read access to the interrupt register. A new arbitration lost interrupt is not possible until the arbitration lost capture register is read out once.

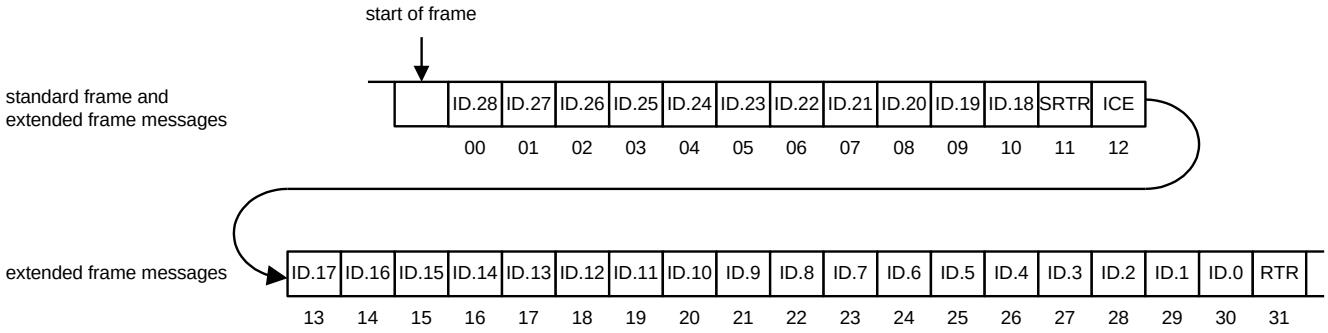


Figure-73. Arbitration Lost Bit Number Interpretation

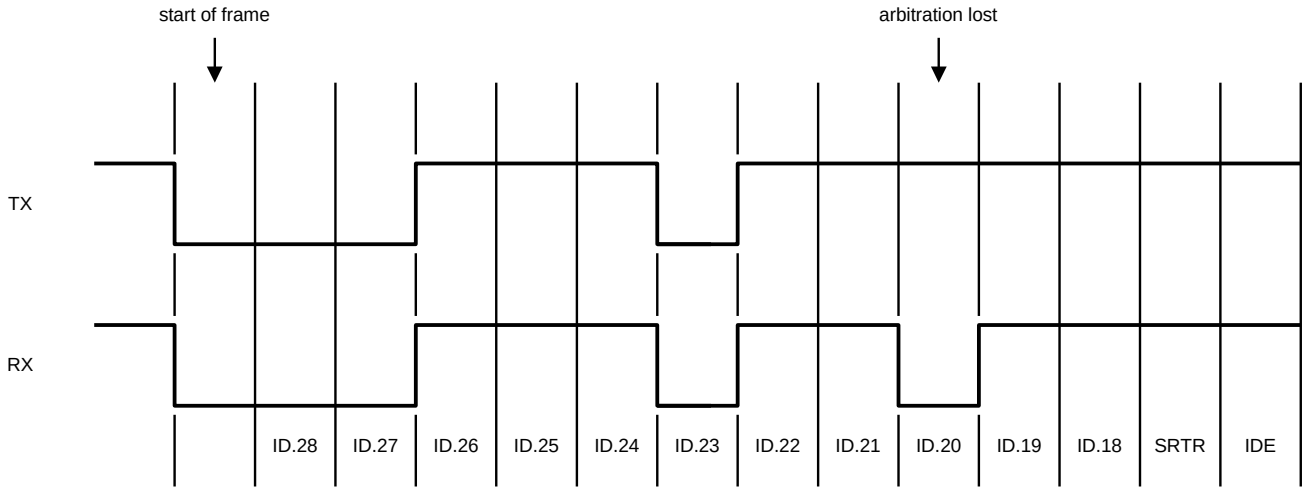


Figure-74. Example 0f arbitration Lost Bit number Interpretation; Result: ALC = 08H

Function of bits 4 to 0 of the Arbitration Lost Capture register

Bits					Decimal Value	Function
ALC.4	ALC.3	ALC.2	ALC.1	ALC.0		
0	0	0	0	0	0	Arbitration lost in bit ID28
0	0	0	0	1	1	Arbitration lost in bit ID27
0	0	0	1	0	2	Arbitration lost in bit ID26
0	0	0	1	1	3	Arbitration lost in bit ID25
0	0	1	0	0	4	Arbitration lost in bit ID24
0	0	1	0	1	5	Arbitration lost in bit ID23
0	0	1	1	0	6	Arbitration lost in bit ID22
0	0	1	1	1	7	Arbitration lost in bit ID21
0	1	0	0	0	8	Arbitration lost in bit ID20
0	1	0	0	1	9	Arbitration lost in bit ID19
0	1	0	1	0	10	Arbitration lost in bit ID18
0	1	0	1	1	11	Arbitration lost in bit SRTR(Note 1)
0	1	1	0	0	12	Arbitration lost in bit IDE
0	1	1	0	1	13	Arbitration lost in bit ID17(Note 2)
0	1	1	1	0	14	Arbitration lost in bit ID16(Note 2)
0	1	1	1	1	15	Arbitration lost in bit ID15(Note 2)
1	0	0	0	0	16	Arbitration lost in bit ID14(Note 2)
1	0	0	0	1	17	Arbitration lost in bit ID13(Note 2)
1	0	0	1	0	18	Arbitration lost in bit ID12(Note 2)
1	0	0	1	1	19	Arbitration lost in bit ID11(Note 2)
1	0	1	0	0	20	Arbitration lost in bit ID10(Note 2)
1	0	1	0	1	21	Arbitration lost in bit ID9(Note 2)
1	0	1	1	0	22	Arbitration lost in bit ID8(Note 2)
1	0	1	1	1	23	Arbitration lost in bit ID7(Note 2)
1	1	0	0	0	24	Arbitration lost in bit ID6(Note 2)
1	1	0	0	1	25	Arbitration lost in bit ID5(Note 2)
1	1	0	1	0	26	Arbitration lost in bit ID4(Note 2)
1	1	0	1	1	27	Arbitration lost in bit ID3(Note 2)
1	1	1	0	0	28	Arbitration lost in bit ID2(Note 2)
1	1	1	0	1	29	Arbitration lost in bit ID1(Note 2)
1	1	1	1	0	30	Arbitration lost in bit ID0(Note 2)
1	1	1	1	1	31	Arbitration lost in bit RTR(Note 2)

Notes:
1. *Bit RTR for standard frame messages.*
2. *Extended frame messages only.*

Error Code Capture Register

Error Code Capture Register (ECC) XFR ADDR: 0B11H (Note 1)

BIT	SYMBOL	NAME	VALUE	FUNCTION
ECC.7	ERRC1	Error code 1	-	Note 2
ECC.6	ERRC0	Error code 0	-	Note 2
ECC.5	DIR	Direction	1	Rx: error occurred during reception
			0	Tx: error occurred during transmission
ECC.4	SEG4	Segment 4	-	Note 3
ECC.3	SEG3	Segment 3	-	
ECC.2	SEG2	Segment 2	-	
ECC.1	SEG1	Segment 1	-	
ECC.0	SEG0	Segment 0	-	

Notes:
1. *If a bus error occurs, the corresponding bus error interrupt is always forced if enabled. At the same time, the current position of the bit stream processor is captured into the Error Code Capture register. The content within this register is fixed until the users software has read out its content once. The capture mechanism is then activated again. The corresponding interrupt flag located in the interrupt register is cleared during the read access to the interrupt register. A new bus error interrupt is not possible until the capture register is read out once.*
2. *For bit interpretation of bits ECC.7 and ECC.6 see table below.*
3. *For bit interpretation of bits ECC.4 to ECC.0 see table below.*

Bit interpretation of bits ECC.7 and ECC.6

ECC.7	ECC.6	FUNCTION
0	0	Bit error
0	1	Form error
1	0	Stuff error
1	1	Other type of error

Bit interpretation of bits ECC.4 to ECC.0

ECC.4	ECC.3	ECC.2	ECC.1	ECC.0	Function
0	0	0	1	1	Start of frame
0	0	0	1	0	ID28 to ID21
0	0	1	1	0	ID20 toID18
0	0	1	0	0	Bit SRTR
0	0	1	0	1	Bit IDE
0	0	1	1	1	ID17 to ID13
0	1	1	1	1	ID12 to ID5
0	1	1	1	0	ID4 to ID0
0	1	1	0	0	Bit RTR

ECC.4	ECC.3	ECC.2	ECC.1	ECC.0	Function
0	1	1	0	1	Reserved bit 1
0	1	0	0	1	Reserved bit 0
0	1	0	1	1	Data length code
0	1	0	1	0	Data field
0	1	0	0	0	CRC sequence
1	1	0	0	0	CRC delimiter
1	1	0	0	1	Acknowledge slot
1	1	0	1	1	Acknowledge delimiter
1	1	0	1	0	End of frame
1	0	0	1	0	Intermission
1	0	0	0	1	Active error flag
1	0	1	1	0	Passive error flag
1	0	0	1	1	Tolerate dominant bits(Note 1)
1	0	1	1	1	Error delimiter
1	1	1	0	0	Overload flag

Note:
1. *Any node tolerates up to 7 consecutive 'dominant' bits after send Active Error Flag, Passive Error Flag or Overload Flag.*

Error Warning Limit Register

Error Warning Limit Register (EWLR) XFR ADDR: 0B12H (Note 1)

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
EWLR.7	EWLR.6	EWLR.5	EWLR.4	EWLR.3	EWLR.2	EWLR.1	EWLR.0

Note:
1. *The Error Warning Limit register can be defined within this register, the default value (after hardware reset) is 96 (Decimal). In Silent mode, this register appears to the CPU as a read/write memory. In Operating mode it is read only. The content change of the EWLR is only possible, if the Silent mode was entered previously. An error status change (see status register) and an error warning interrupt forced by the new register content will not occur until the Silent mode is quit again.*

Receive Error Counter Register

Receive Error Counter Register (RXERR) XFR ADDR: 0B13H (Note 1)

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
RXERR.7	RXERR.6	RXERR.5	RXERR.4	RXERR.3	RXERR.2	RXERR.1	RXERR.0

Note:
1. *The Receive Error Counter register reflects the current value of the receive error counter. After hardware reset, this register is initialized to logic 0. In operating mode this register appears to the CPU as a read only memory. A write access to this register is possible only in Silent mode. If a bus-off event occurs, the receive error counter is initialized to logic 0. The time bus-off is valid, writing to this register has no effect. Note that a CPU-forced content change of the receive error counter is only possible, if the Silent mode was entered previously. An error status change (see status register), an error warning or an error passive interrupt forced by the new register content will not occur, until the Silent mode is cancelled again.*

Transmit Error Counter Register

Transmit Error Counter Register (TXERR) XFR ADDR: 0B14H (Note 1)

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
TXERR.7	TXERR.6	TXERR.5	TXERR.4	TXERR.3	TXERR.2	TXERR.1	TXERR.0

Note:
1. *The Transmit Error Counter register reflects the current value of the transmit error counter.*
2. *In operating mode this register appears to the CPU as a read only memory. A write access to this register is possible only in Silent mode. After hardware reset, this register is initialized to logic 0. If a bus-off event occurs, the transmit error counter is initialized to 128.*

Received Frame Counter Register

Received Frame Counter Register (RFCR) XFR ADDR: 0B15H (Note 1)

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
RFCR.7 (Note 2)	RFCR.6	RFCR.5	RFCR.4	RFCR.3	RFCR.2	RFCR.1	RFCR.0

Note:
1. *When CAN controller enters into Silent mode, the register is cleared. In Operating mode, the register is read only. Value of the register will be added by 1, if a frame is received successfully. The register will count down when 'Release Receive Buffer 'command bit is set logic 1.*
2. *RFCR.7 is a reserved bit and shall be zero when read this bit.*

Received Frame Information Register

Received Frame Information Register (RIFR) XFR ADDR: 0B16H (Note 1)

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
0	RIFR.6 (Note2)	0	Receive Filter Number (Note 3)				

Note:
1. *This register is read only.*
2. *RIFR.6 is set when the Receive ID matches the Receive Filter Number and its Forward Enable Register (XFR 0X0B05~0X0B08) was set.*
3. *RIFR[4:0] indicates the Receive Filter Number that the received data passed through.*

Transmit Buffer

The global layout of the transmit buffer is shown in Figure below. One has to distinguish between the Standard Frame Format (SFF) and the Extended Frame Format (EFF) configuration. The transmit buffer allows the definition of one transmit message with up to eight data bytes.

Standard Format Frame

CAN Address	B17h	TX Frame Information
	B18h	TX Identifier 4
	B19h	TX Identifier 3
	B1Ah	TX Identifier 2
	B1Bh	TX Identifier 1
	B1Ch	TX Data 1
	B1Dh	TX Data 2
	B1Eh	TX Data 3
	B1Fh	TX Data 4
	B20h	TX Data 5
	B21h	TX Data 6
	B22h	TX Data 7
	B23h	TX Data 8

Extended Format Frame

CAN Address	B17h	TX Frame Information
	B18h	TX Identifier 4
	B19h	TX Identifier 3
	B1Ah	TX Identifier 2
	B1Bh	TX Identifier 1
	B1Ch	TX Data 1
	B1Dh	TX Data 2
	B1Eh	TX Data 3
	B1Fh	TX Data 4
	B20h	TX Data 5
	B21h	TX Data 6
	B22h	TX Data 7
	B23h	TX Data 8

Figure-75. Transmit Buffer Layout For Standard And Extended Frame Format Configurations

The transmit buffer layout is subdivided into descriptor and data fields where the first byte of the descriptor field is the frame information byte (frame information). It describes the frame format (SFF or EFF), remote or data frame and the data length. Four identifier bytes for both SFF and EFF messages follow. But only the lower 11 bits in the four identifier bytes are valid for SFF and the upper 21 bits are invalid. For EFF, the lower 29 bits in the four identifier bytes are valid and the upper 3 bits are invalid. The data field contains up to eight data bytes.

The transmit buffer has a length of 13 bytes. Note that a direct access to the transmit buffer RAM is possible using the XFR address from 0X0B17 to 0X0B23. Register mapping 0X0B17 to 0X0B23 should be programmed in operation mode. This RAM area is reserved for the transmit buffer.

Transmit Frame Information Register

Transmit Frame Information Register XFR ADDR: 0B17H

BIT	SYMBOL	NAME	VALUE	FUNCTION
TXFIR.7	FF	Frame format	1	Extended format frame
			0	Standard format frame
TXFIR.6	RTR	Remote frame transmit request	1	Remote frame
			0	Data frame
TXFIR.5	Reserved	Reserved	0	Reserved
TXFIR.4	Reserved	Reserved	0	Reserved
TXFIR.3	DLC.3	Data length code 3	Note 1	
XFIR.2	DLC.2	Data length code 2		
TXFIR.1	DLC.1	Data length code 1		
TXFIR.0	DLC.0	Data length code 0		

Notes:

1. *The number of bytes in the data field of a message is coded by the data length code. At the start of a remote frame transmission the data length code is not considered due to the RTR bit being logic 1 (remote). This forces the number of transmitted/received data bytes to be 0. Nevertheless, the data length code must be specified correctly to avoid bus errors, if two CAN controllers start a remote frame transmission with the same identifier simultaneously. The range of the data byte count is 0 to 8 bytes and is coded as follows:*

$$\text{DataByteCount} = 8 \times DLC.3 + 4 \times DLC.2 + 2 \times DLC.1 + DLC.0$$

For reasons of compatibility, no data length code >8 should be used. If a value >8 is selected, 8 bytes are transmitted in the data frame with the Data Length Code specified in DLC.

Identifier Register

In standard Frame Format (SFF) the identifier consists of 11 bits (ID.28 to ID.18) and in Extended Frame Format (EFF) messages the identifier consists of 29 bits (ID.28 to ID.0). ID.28 is the most significant bit, which is transmitted first on the bus during the arbitration process. The identifier acts as the message's name, used in a receiver for acceptance filtering, and also determines the bus access priority during the arbitration process. The smaller the binary value of the identifier is, the higher the priority is. This is due to the larger number of leading dominant bits during arbitration.

Standard Format Frame Identifier

TX Identifier Register 3 XFR ADDR：0B18H

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
X (Note 1)	X (Note 1)	X (Note 1)	X (Note 1)	X (Note 1)	X (Note 1)	X (Note 1)	X (Note 1)

TX Identifier Register 2 XFR ADDR: 0B19H

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
X (Note 1)	X (Note 1)	X (Note 1)	X (Note 1)	X (Note 1)	X (Note 1)	X (Note 1)	X (Note 1)

TX Identifier Register 1 **XFR ADDR: 0B1AH**

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
X (Note 1)	X (Note 1)	X (Note 1)	X (Note 1)	X (Note 1)	ID.28 (Note 2)	ID.27 (Note 2)	ID.26 (Note 2)

TX Identifier Register 0 **XFR ADDR: 0B1BH**

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
ID.25 (Note 1)	ID.24 (Note 1)	ID.23 (Note 1)	ID.22 (Note 1)	ID.21 (Note 1)	ID.20 (Note 1)	ID.19 (Note 1)	ID.18 (Note 1)

Note:
1. ID.X means identifier bit X

Extended Format Frame Identifier

TX Identifier Register **XFR ADDR: 0B18H**

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
X (Note 1)	X (Note 1)	X (Note 1)	ID.28 (Note 2)	ID.27 (Note 2)	ID.26 (Note 2)	ID.25 (Note 2)	ID.24 (Note 2)

Notes:
1. Don't care
2. ID.X means identifier bit X

TX Identifier Register 2 **XFR ADDR: 0B19H**

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
ID.23 (Note 1)	ID.22 (Note 1)	ID.21 (Note 1)	ID.20 (Note 1)	ID.19 (Note 1)	ID.18 (Note 1)	ID.17 (Note 1)	ID.16 (Note 1)

TX Identifier Register 1 **XFR ADDR: 0B1AH**

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
ID.15 (Note 1)	ID.14 (Note 1)	ID.13 (Note 1)	ID.12 (Note 1)	ID.11 (Note 1)	ID.10 (Note 1)	ID.9 (Note 1)	ID.8 (Note 1)

Note:
1. ID.X means identifier bit X

TX Identifier Register 0 **XFR ADDR: 0B1BH**

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
ID.7 (Note 1)	ID.6 (Note 1)	ID.5 (Note 1)	ID.4 (Note 1)	ID.3 (Note 1)	ID.2 (Note 1)	ID.1 (Note 1)	ID.0 (Note 1)

Note:
1. ID.X means identifier bit X

Data field

Data field registers is mapping to XFR address 0B1CH to 0B23H.
The number of transferred data bytes is defined by the data length code. The first bit transmitted is the most significant bit of data byte 1 at XFR address 0B1CH.

Filter Status Registers

Filter Status Register 3(FSR3) XFR ADDR: 0B24H (Note 1)

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
FSR3.7	FSR31	Filter 31 matching status	0	1	Filter 31 is matched
				0	Filter 31 is not matched
FSR3.6	FSR30	Filter 30 matching status	0	1	Filter 30 is matched
				0	Filter 30 is not matched
FSR3.5	FSR29	Filter 29 matching status	0	1	Filter 29 is matched
				0	Filter 29 is not matched
FSR3.4	FSR28	Filter 28 matching status	0	1	Filter 28 is matched
				0	Filter 28 is not matched
FSR3.3	FSR27	Filter 27 matching status	0	1	Filter 27 is matched
				0	Filter 27 is not matched
FSR3.2	FSR26	Filter 26 matching status	0	1	Filter 26 is matched
				0	Filter 26 is not matched
FSR3.1	FSR25	Filter 25 matching status	0	1	Filter 25 is matched
				0	Filter 25 is not matched
FSR3.0	FSR24	Filter 24 matching status	0	1	Filter 24 is matched
				0	Filter 24 is not matched

Note:

1. *This is a read-only register in Operating mode. Reading or writing operation in Silent mode is for Dispatch time 3 register. When a frame message can pass through a filter, the respective filter matching status bit is set to logic 1.*

Filter Status Register 2(FSR2) XFR ADDR: 0B25H (Note 1)

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
FSR2.7	FSR23	Filter 23 matching status	0	1	Filter 23 is matched
				0	Filter 23 is not matched
FSR2.6	FSR22	Filter 22 matching status	0	1	Filter 22 is matched
				0	Filter 22 is not matched
FSR2.5	FSR21	Filter 21 matching status	0	1	Filter 21 is matched
				0	Filter 21 is not matched
FSR2.4	FSR20	Filter 20 matching status	0	1	Filter 20 is matched
				0	Filter 20 is not matched
FSR2.3	FSR19	Filter 19 matching status	0	1	Filter 19 is matched
				0	Filter 19 is not matched

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
FSR2.2	FSR18	Filter 18 matching status	0	1	Filter 18 is matched
				0	Filter 18 is not matched
FSR2.1	FSR17	Filter 17 matching status	0	1	Filter 17 is matched
				0	Filter 17 is not matched
FSR2.0	FSR16	Filter 16 matching status	0	1	Filter 16 is matched
				0	Filter 16 is not matched

Note:
1. *This is a read-only register in operating mode. Reading or writing operation is for Dispatch time 2 register. When a frame message can pass through a filter, the respective filter matching status bit is set to logic* 1.

Filter Status Register 1(FSR1) XFR ADDR: 0B26H (Note 1)

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
FSR1.7	FSR15	Filter 15 matching status	0	1	Filter 15 is matched
				0	Filter 15 is not matched
FSR1.6	FSR14	Filter 14 matching status	0	1	Filter 14 is matched
				0	Filter 14 is not matched
FSR1.5	FSR13	Filter 13 matching status	0	1	Filter 13 is matched
				0	Filter 13 is not matched
FSR1.4	FSR12	Filter 12 matching status	0	1	Filter 12 is matched
				0	Filter 12 is not matched
FSR1.3	FSR11	Filter 11 matching status	0	1	Filter 11 is matched
				0	Filter 11 is not matched
FSR1.2	FSR10	Filter 10 matching status	0	1	Filter 10 is matched
				0	Filter 10 is not matched
FSR1.1	FSR9	Filter 9 matching status	0	1	Filter 9 is matched
				0	Filter 9 is not matched
FSR1.0	FSR8	Filter 8 matching status	0	1	Filter 8 is matched
				0	Filter 8 is not matched

Note:
1. *This is a read-only register in operating mode. Reading or writing operation is for Dispatch time 1 register. When a frame message can pass through a filter, the respective filter matching status bit is set to logic 1.*

Filter Status Register 0(FSR0) **XFR ADDR: 0B27H (Note 1)**

BIT	SYMBOL	NAME	Reset Value	VALUE	FUNCTION
FSR0.7	FSR7	Filter 7 matching status	0	1	Filter 7 is matched
				0	Filter 7 is not matched
FSR2.6	FSR6	Filter 6 matching status	0	1	Filter 6 is matched
				0	Filter 6 is not matched
FSR0.5	FSR5	Filter 5 matching status	0	1	Filter 5 is matched
				0	Filter 5 is not matched
FSR0.4	FSR4	Filter 4 matching status	0	1	Filter 4 is matched
				0	Filter 4 is not matched
FSR0.3	FSR3	Filter 3 matching status	0	1	Filter 3 is matched
				0	Filter 3 is not matched
FSR0.2	FSR2	Filter 2 matching status	0	1	Filter 2 is matched
				0	Filter 2 is not matched
FSR0.1	FSR1	Filter 1 matching status	0	1	Filter 1 is matched
				0	Filter 1 is not matched
FSR0.0	FSR0	Filter 0 matching status	0	1	Filter 0 is matched
				0	Filter 0 is not matched

Note:
1. *This is a read-only register in operating mode. Reading or writing operation is for Dispatch time 0 register. When a frame message can pass through a filter, the respective filter matching status bit is set to logic 1.*

Dispatch Time Registers

Dispatch Time registers contains 4 dispatch enable bits and 28 Dispatch Time bits. When the 28 dispatch time bits are all set to logic 0, the dispatch function is disabled.

Dispatch Time 3 Register (DTR3) **XFR ADDR: 0B24H (Note 1)**

DTR3.7	DTR3.6	DTR3.5	DTR3.4	DTR3.3	DTR3.2	DTR3.1	DTR3.0
WP3 (Note 2)	WP2 (Note 2)	WP1 (Note 2)	WP0 (Note 2)	DTR27 (Note 3)	DTR26 (Note 3)	DTR25 (Note 3)	DTR24 (Note 3)

Note:
1. *After hardware reset, the value of the register is 0.*
2. *The four bits are write-protect bits and can be written only in Silent mode. Only when {WP3, WP2, WP1, WP0} is 0X6, MCU can write DTR2, DTR1 and DTR0. Or the value written to the three registers is invalid.*
3. *The upper 4 dispatch time bits. The four bits can be written only in Silent mode.*

Dispatch Time 2 Register (DTR2) **XFR ADDR: 0B25H (Note 1)**

DTR2.7	DTR2.6	DTR2.5	DTR2.4	DTR2.3	DTR2.2	DTR2.1	DTR2.0
DTR23	DTR22	DTR21	DTR20	DTR19	DTR18	DTR17	DTR16

Note:
1. *The dispatch time 2 register can be written only in Silent mode and the value of write-protect bits is 0X6. After hardware reset, the value of the register is 0.*

Dispatch Time 1 Register (DTR1) XFR ADDR: 0B26H (Note 1)

DTR1.7	DTR1.6	DTR1.5	DTR1.4	DTR1.3	DTR1.2	DTR1.1	DTR1.0
DTR15	DTR14	DTR13	DTR12	DTR11	DTR10	DTR9	DTR8

Note:
1. *The dispatch time 1 register can be written only in Silent mode and the value of write-protect bits is 0X6. After hardware reset, the value of the register is 0.*

Dispatch Time 0 Register (DTR0) XFR ADDR: 0B27H (Note 1)

DTR0.7	DTR0.6	DTR0.5	DTR0.4	DTR0.3	DTR0.2	DTR0.1	DTR0.0
DTR7	DTR6	DTR5	DTR4	DTR3	DTR2	DTR1	DTR0

Note:
1. *The dispatch time 0 register can be written only in Silent mode and the value of write-protect bits is 0X6. After hardware reset, the value of the register is 0.*

The least significant bit equal to 200uS
If dispatch period is 50 mS, adjust DTR0 (0XB27) to be 250
Then 250 * 200 uS=50,000 uS = 50 ms
The dispatch time will be 50ms
These four registers 0XB24 to 0XB27 only can program in silent mode.

26.3 Basic Function

The basic function of CAN controller will be discussed in this section. The following chapter will describe the advanced functions of CAN controller.

To be communication with other node in the CAN-bus, CAN controller must be configured properly as follows:

26.3.1 Pin Configuration

In CS8959, each CAN controller has two groups of pins. We can configure the pins by the register in the XFR address 0XF052. Note that we cannot validate the two groups of pins at the same time for each CAN controller.

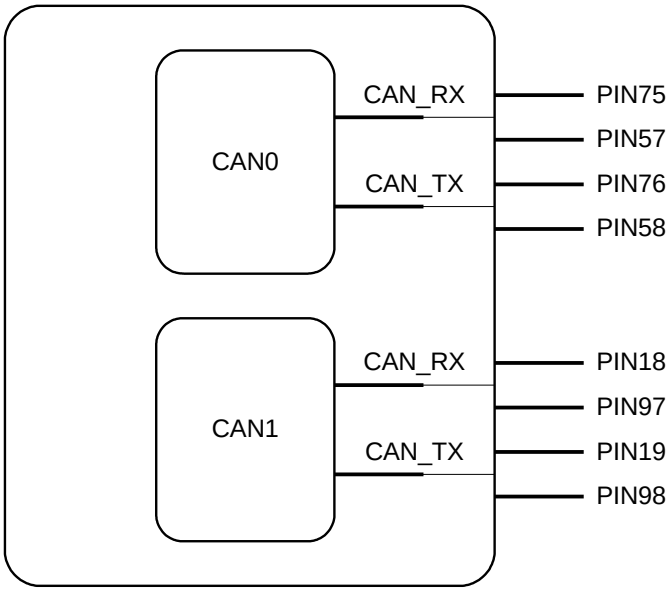


Figure-76. CAN TX pin and RX pin distribution

CAN Pins Configuration Register **XFR ADDR: 0F52H**

BIT3	BIT2	BIT1	BIT0
1 : PIN#75 = CAN0_RX PIN#76 = CAN0_TX 0: PIN#75 = GPIO P40 PIN#76 = GPIO P41	1 : PIN#57 = CAN0_RX PIN#58 = CAN0_TX 0: PIN#57 = GPIO P80 PIN#58 = GPIO P81	1 : PIN#18 = CAN1_RX PIN#19 = CAN1_TX 0: PIN#18 = GPIO P42 PIN#19 = GPIO P43	1 : PIN#97 = CAN1_RX PIN#98 = CAN1_TX 0: PIN#97 = GPIO P20 PIN#98 = GPIO P21

For example, we can write the value: 0X0A, 0X09, 0X06 or 0X05 into the lower four bits into the CAN Pins Configuration register. The configurable pins supply a group of backup pins to avoid embarrassment in case of transceiver failure.

26.3.2 Proper setting of Operating mode and Silent mode

Some registers are writable only in Silent mode such as Bus Timing registers, so we must write these registers in the Silent mode and the CAN controller must quit Silent mode when these registers are configured properly to make the controller work (see also the “Mode Register ” in chapter “Register Description”).

26.3.3 Proper setting of Baud Rate

Two registers BTR0 and BTR1 determine the baud rate in the CAN-bus. For detailed information about the two registers, please refer to section “Bus Timing Register” in the chapter “Register Description”.
The following example explains how to calculate the baud rate:

Supposed: the oscillator frequency is 16MHZ,value of the BTR0 and BTR1 is 01h and 14h, and so we can get the baud rate 500kbps as the following way:

$$BRP = 32\times BRP.5+16\times BRP.4+8\times BRP.3+4\times BRP.2+2\times BRP.1+BRP.0+1=1$$

$$t_{scl} = 2\times t_{clk}\times(BRP+1) = \frac{2\times 2}{16\times 10^{6}} = 0.25\times 10^{-6}s$$

$$t_{SYNCSEG} = 1\times t_{scl} = 0.25\times 10^{-6}s$$

$$t_{SEG1} = t_{scl}\times(8\times TSEG1.3+4\times TSEG1.2+2\times TSEG1.1+TSEG1.0+1) = 5\times t_{scl} = 1.25\times 10^{-6}s$$

$$t_{SEG2} = t_{scl}\times(4\times TSEG2.2+2\times TSEG2.1+TSEG2.0+1) = 2\times t_{scl} = 0.5\times 10^{-6}s$$

$$BT = t_{SYNCSEG}+t_{SEG1}+t_{SEG2} = 2\times 10^{-6}s$$

BT stands for the time that a single bit lasts.

So the baud rate in the CAN-bus is $\frac{1}{BT} = \frac{1}{2\times 10^{-6}} = 500\times 10^{3}bps = 500kbps$

26.3.4 The setting of the acceptance filter

With the help of the acceptance filter, the CAN controller can receive proper frames from CAN-bus only when the identifier bits of the received frames are equal to the predefined ones within the acceptance filter registers.

The acceptance filter is defined by the Acceptance Code Registers (ACR) and the Mask Code Registers (MCR). The bit patterns of frames to be received are defined within the Acceptance Code registers. The corresponding mask code registers allow define certain bit positions to be 'don't care'. After hardware reset, all the Acceptance Code registers and Mask Code registers are cleared.

In CS8959, each CAN controller has 32 groups of acceptance filter. We must enable at least one group of filter previously. Each group of filter has 32 Acceptance Code bits and 32 Mask Code bits and occupies 8 bytes address space. They are mapping to the XFR address space as the Figures below. Acceptance code 3 register is the highest byte in the Acceptance Code; the Acceptance code 2 register is the lowest byte in the Acceptance Code. Mask code 3 register is the highest byte in the Mask Code and Mask Code 0 register is the lowest byte in the Mask Code. The Acceptance Code Register and Mask Code Register distribute crossly in the XFR address space. We take Filter 0 in CAN0 as an example: the address of the Acceptance Code 3 is 0X0B28 and the Mask Code 3 is 0X0B29. The address of the Acceptance Code 2 is 0X0B2A and the Mask Code 2 is 0XB2B and so on. When received message is standard format frame, the lower 11 bits of the Acceptance Code and the Mask Code are valid. When received message is extended format frame, the lower 29 bits of the Acceptance Code and the Mask Code are valid.

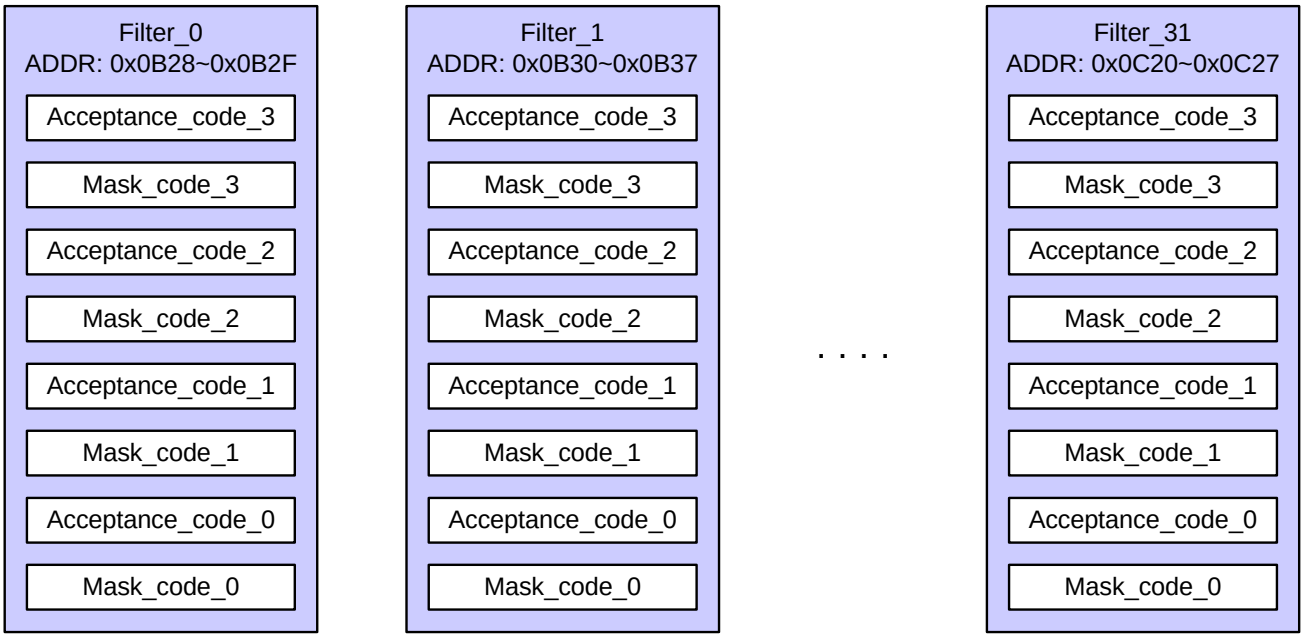


Figure-77. CAN0 Acceptance Filters

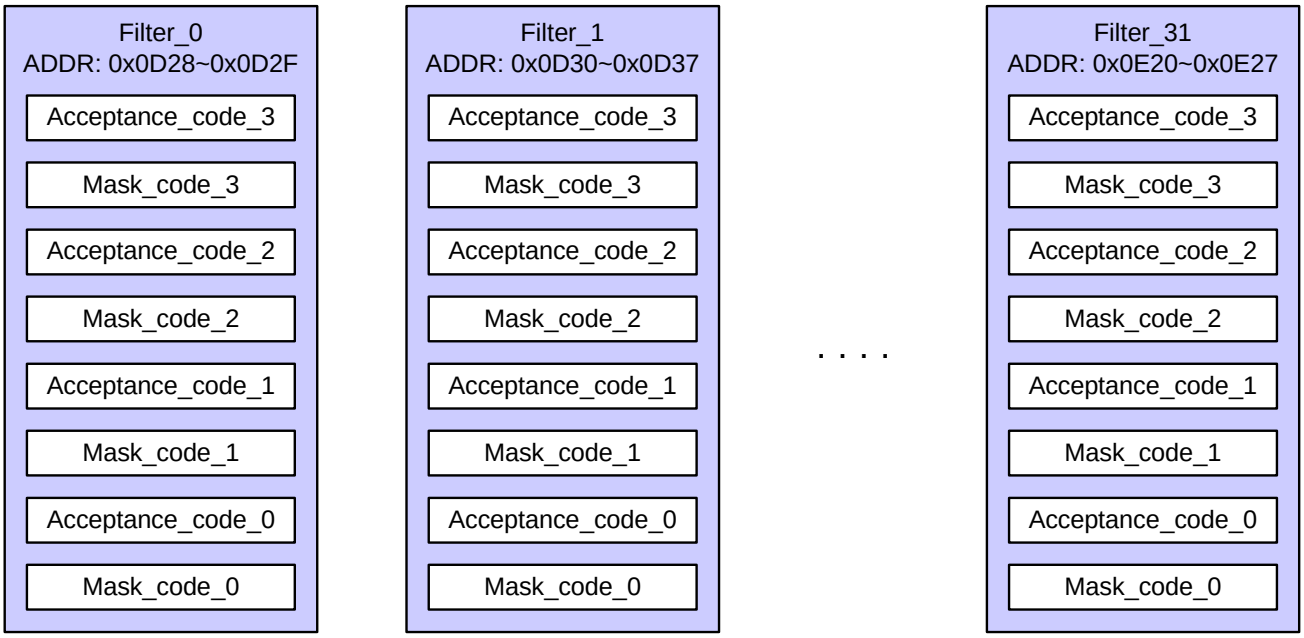


Figure-78. CAN1 Acceptance Filters

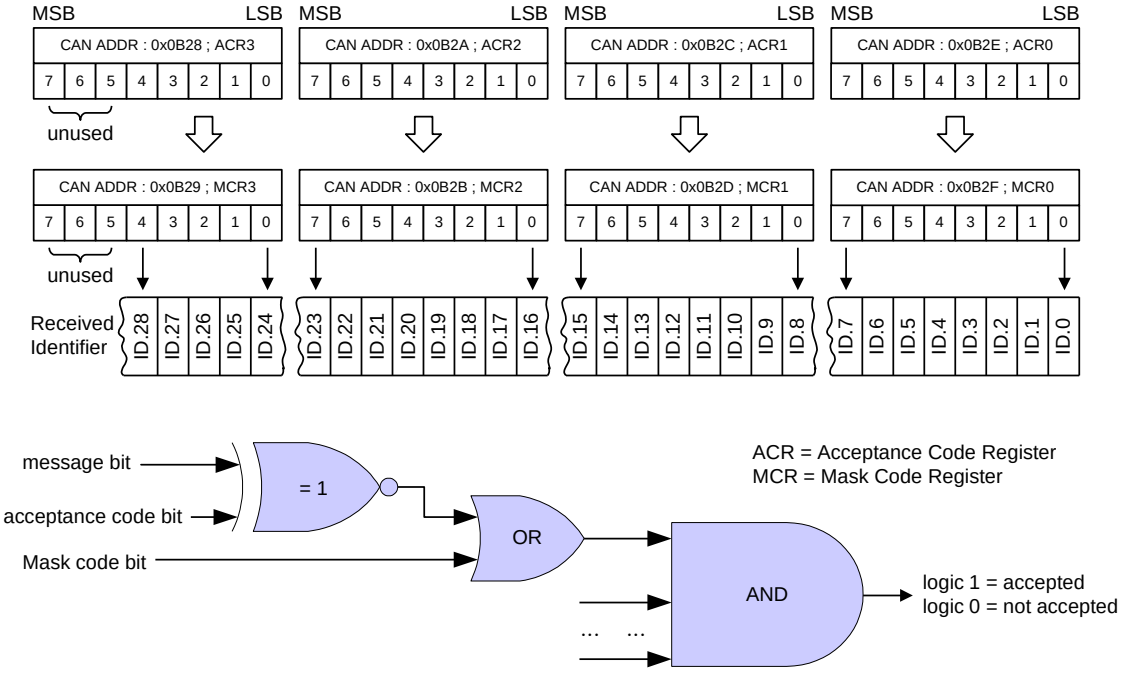


Figure-79. Filter 0 In CAN0 Is Receiving Extended Frame Messages

Extended frame: if an extended format frame is received, the complete identifier including the RTR bit is used for acceptance filtering. For a successful reception of a message, all single bit comparisons have to signal acceptance. It should be noted that the 3 most significant bits of ACR3 and MCR3 are not used. In order to be compatible with future products these bit should be ‘don’t care’ by setting MCR3.7, MCR3.6 and MCR3.5 to logic 1.

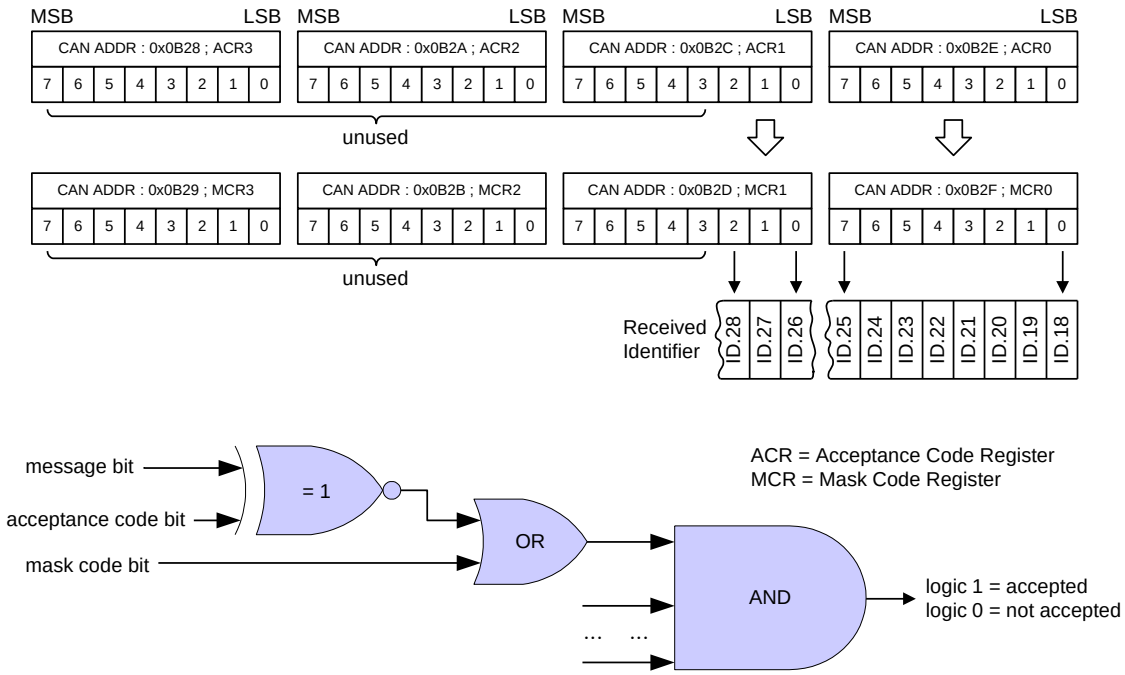


Figure-80. Filter 0 in CAN0 Is Receiving Standard Frame Messages

Standard frame: if a standard format frame message is received, the completely identifier are used for acceptance filtering. For a successful reception of a frame, all single bit comparisons have to signal acceptance. Note that ACR3, MCR3, ACR2, MCR2 and the 5 most significant bits of ACR1 and MCR1 are not used. In order to be compatible with future products, these unused bits in Mask Code registers should be programmed to be ‘don’t care’ by setting these bits to logic 1.

26.3.5 The setting of Interrupt

There are seven interrupt sources, which can be responded by MCU. Three interrupt vectors CAN0I, CAN1I and CANBUSI are attributed to the two CAN controllers, CAN0I to CAN0, CAN1I to CAN1 and CANBUSI to both controllers. When respective interrupt is enabled, CAN0I can respond the seven interrupt sources' request in CAN0 and CAN1I can respond the seven interrupt sources' request in CAN1. CANBUSI can respond when at least one controller enters into bus off.

CAN0I, CAN1I and CANBUSI

Interrupt Flag	Active level/edge	Flag resets	Vector
CAN0I	Low	Hardware	0X4B
CAN1I	Low	Hardware	0X53
CANBUSI	Falling	Software	0X73

For detailed information about the CAN0I, CAN1I and CANBUSI, please refer to the 'Interrupt System'.

26.3.6 Transmitting a frame

You must write transmit buffer to send a frame to CAN-bus, then set the transmit request bit to logic 1. The controller will wait for the bus idle and sends the frame to the bus.

For example, CAN0 will send a standard format frame to CAN-bus. Frame identifier is 0X001, the data have a length of 8 and the data are 11h, 22h, 33h, 44h, 55h, 66h, 77h and 88h..The program written in Keil-C is as follows:

```
Write_reg(0X0B17,0X08);  //write the frame information register, FF=0,  RTR=0,  DLC=8
Write_reg(0X0B18,0X00);  //write the TX ID3 Register
Write_reg(0X0B19,0X00);  // write the TX ID2 Register
Write_reg0X0B1A,0X00);  // write the TX ID1 Register
Write_reg(0X0B1B,0X01);  // write the TX ID0 Register
Write_reg(0X0B1C,0X11);  //write the TX data 1 Register
Write_reg(0X0B1C,0X22);  // write the TX data 2 Register
Write_reg0X0B1C,0X33);   // write the TX data 3 Register
Write_reg(0X0B1C,0X44);  // write the TX data 4 Register
Write_reg(0X0B1C,0X55);  // write the TX data 5 Register
Write_reg(0X0B1C,0X66);  // write the TX data 6 Register
Write_reg(0X0B1C,0X77);  // write the TX data 7 Register
Write_reg(0X0B1C,0X88);  // write the TX data 8 Register
Write_reg(0X0B01,0X01);   //write the transmit request bit
```

Note that the prototype of the function Write_reg is :

```
Void Write_reg(unsigned int address, unsigned char value );
```

26.3.7 Receiving a frame

When CAN controller receives a frame successfully and the receive interrupt is enabled, the receive interrupt bit will be set logic 1. MCU is informed by the receive interrupt and reads the data in the receive buffer out, then releases the receive buffer. Please pay special attention here that the data must be read before releasing receive buffer, because the value in the receive buffer has changed after releasing receive buffer.

26.4 Advanced Function

The CAN controller in CS8959 also has some distinct functions as follows:

26.4.1 Dispatch

The CAN controller can transmit data periodically to CAN-bus to report its status. To realize this function, we must configure the dispatch time registers properly. The 28 dispatch time bits control the period that the controller transmits data to CAN-bus and the dispatch period is calculate as follows:

Provided the bit period is $t_b = \frac{1}{BaudRate}$ and the value of the 28 dispatch time bits is "A", then the dispatch time is $100 \times A \times t_b$ second. In other words, the CAN controller transmits a message in the TX buffer to the CAN-bus every $100 \times A \times t_b$ second. When dispatch time bits are all set to logic 0, the dispatch function is disabled. When bus error in a dispatch frame, it can transmit an error frame and can't transmit the failed frame again.

26.4.2 Forward

Forward is a very important function for gateway. The following will illustrate the forward function. CAN0 is connected with net A and CAN1 is connected with net B. The Baud Rate in net A and net B may be different. The data received from net A by CAN0 can be transmitted to net B by CAN1.

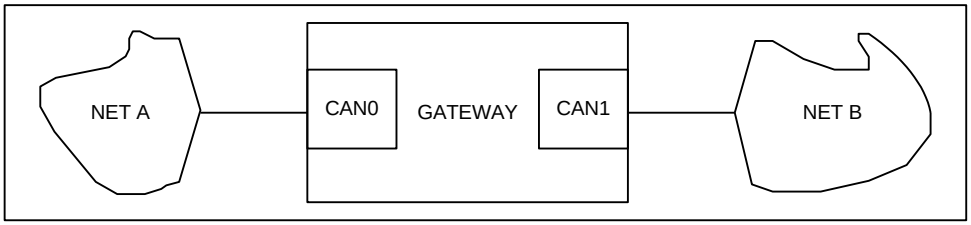


Figure-81. Different Nets Connected By CAN Gateway In CS8959

The following steps illustrate how to realize CAN1 transmitting the data received by CAN0:

1. Set the Mask Code registers and Acceptance Code registers in CAN0 to receive proper frames. If we want CAN1 to forward the data with ID 0X00000001 received by CAN0, we can write proper value into Mask Code register and Acceptance Code register of filer 0 in CAN0:
   XFR[0X0B28]=00h; XFR[0X0B2A]=00h; XFR[0X0B2C]=00h; XFR[0X0B2E]=01h;
   XFR[0X0B29]=00h; XFR[0X0B2B]=00h; XFR[0X0B2D]=00h; XFR[0X0B2F]=00h;
2. Set the Forward enable registers in CAN0 to set the Forward Flag of the received frame to logic 1. We can set the FWE0 bit to logic 1 to make filter 0 forward enabled.
3. Set the Forward Request bit in Command Register of CAN0 to logic 1 and CAN1 Transmits a frame in CAN0's RX buffer to net B.

The data received from net B by CAN1 also can be transmitted to net A by CAN0.

The following steps illustrate how to realize CAN0 transmitting the data received by CAN1:

1. Set the Mask Code registers and Acceptance Code registers in CAN1 to receive proper frames. If we want CAN0 to forward the data with ID 0X00000001 received by CAN1, we can write proper value into Mask Code register and Acceptance Code register of filer 0 in CAN1:
   XFR[0X0B28]=00h; XFR[0X0B2A]=00h; XFR[0X0B2C]=00h; XFR[0X0B2E]=01h;
   XFR[0X0B29]=00h; XFR[0X0B2B]=00h; XFR[0X0B2D]=00h; XFR[0X0B2F]=00h;
2. Set the Forward enable registers in CAN1 to set the Forward Flag of the received frame to logic 1. We can set the FWE0 bit to logic 1 to make filter 0 forward enabled.
3. Set the Forward Request bit in Command Register of CAN1 to logic 1 and CAN1 Transmits a frame in CAN0's RX buffer to net B.

26.5 CAN Test

We provide two test modes: self loop test and dual loop test. Both the two test modes support online test and the transceiver needn't participate in the two test modes.

26.5.1 Self-Loop Test

CAN0 In Self-Loop Test

The following figure illustrates the internal connection when CAN0 is in self-loop test mode. Self-loop test usually goes on in Self Test mode (see Mode register) with Self Receive command to test the controller whether it works. When going on self-loop test, the controller can't receive data from CAN-bus.

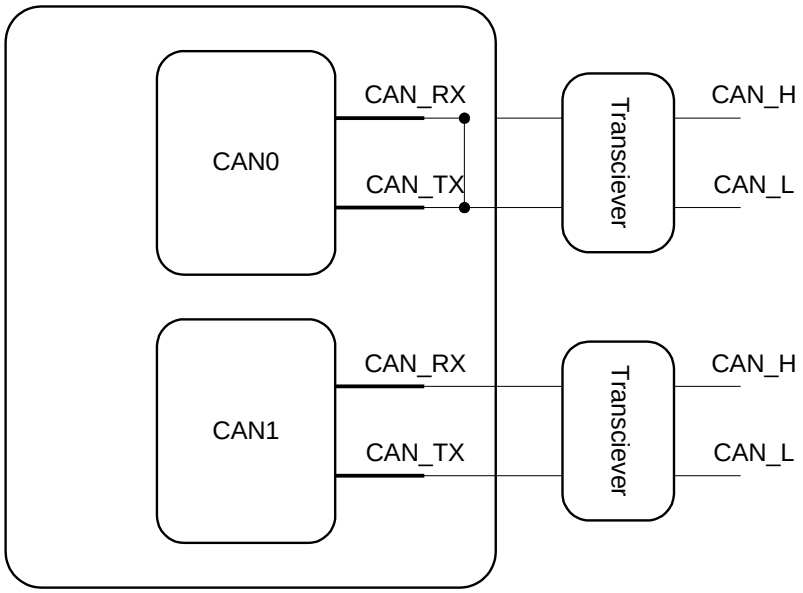


Figure-82. CAN0 In Self-Loop Test

CAN1 in self-loop test

The following figure illustrates the internal connection when CAN1 is in self-loop test mode. Self-loop test usually goes on in Self Test mode (see Mode register) with Self Receive command to test the controller whether it works. When going on self-loop test, the controller can't receive data from CAN-bus.

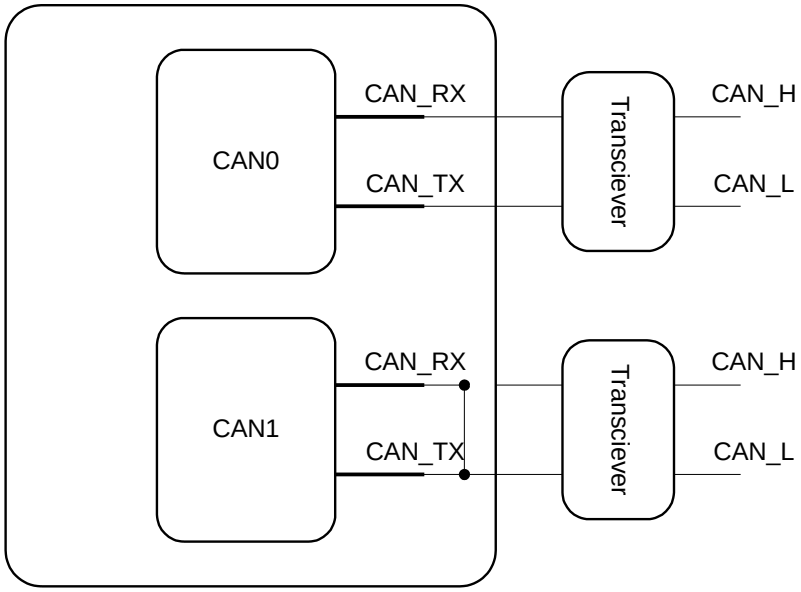


Figure-83. CAN1 In Self-loop Test

26.5.2 Dual loop test

The following figure illustrates the internal connection when the two controllers are in dual loop test. When going on dual loop test, the two controllers can't receive data from CAN-bus and the two controllers communication with each other.

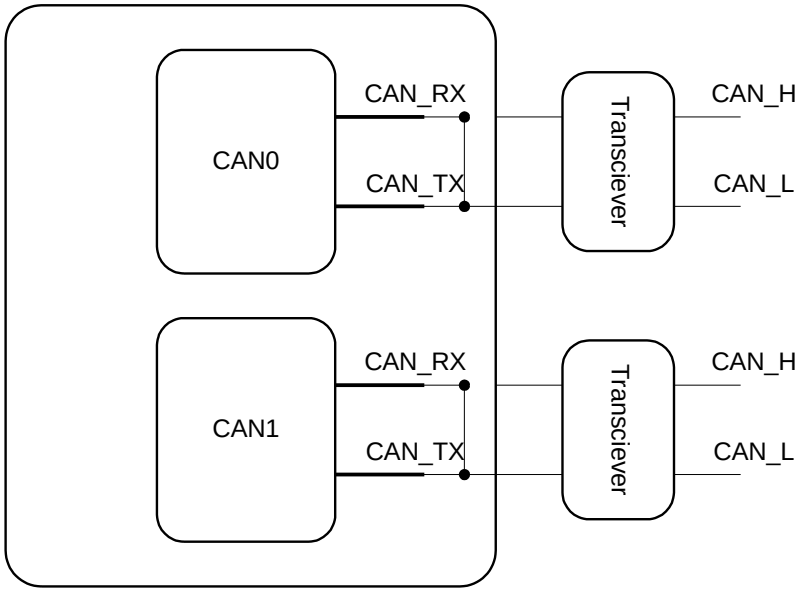


Figure-84. Dual Loop Test

27. ELECTRICAL CHARACTERISTICS

27.1 Absolute Maximum Rating

SYMBOL	PARAMETER	RATING	UNIT
VDD	Positive Power Supply	4.5 ~ 5.5	V
T_A	Ambient Operating Temperature	-40 to 85	^oC
T_{STG}	Storage Temperature	-65 to 150	^oC
RTC_{VDD}	RTC Positive Power Supply	1.1 to 2.75	V

27.2 Recommend Operating Condition

SYMBOL	PARAMETER	RATING	UNIT
VDD	Positive Power Supply	5	V
T_A	Ambient Operating Temperature	-40 to 85	^oC
RTC_{VDD}	RTC Positive Power Supply	2.5	V

27.3 DC/AC Electrical Characteristics

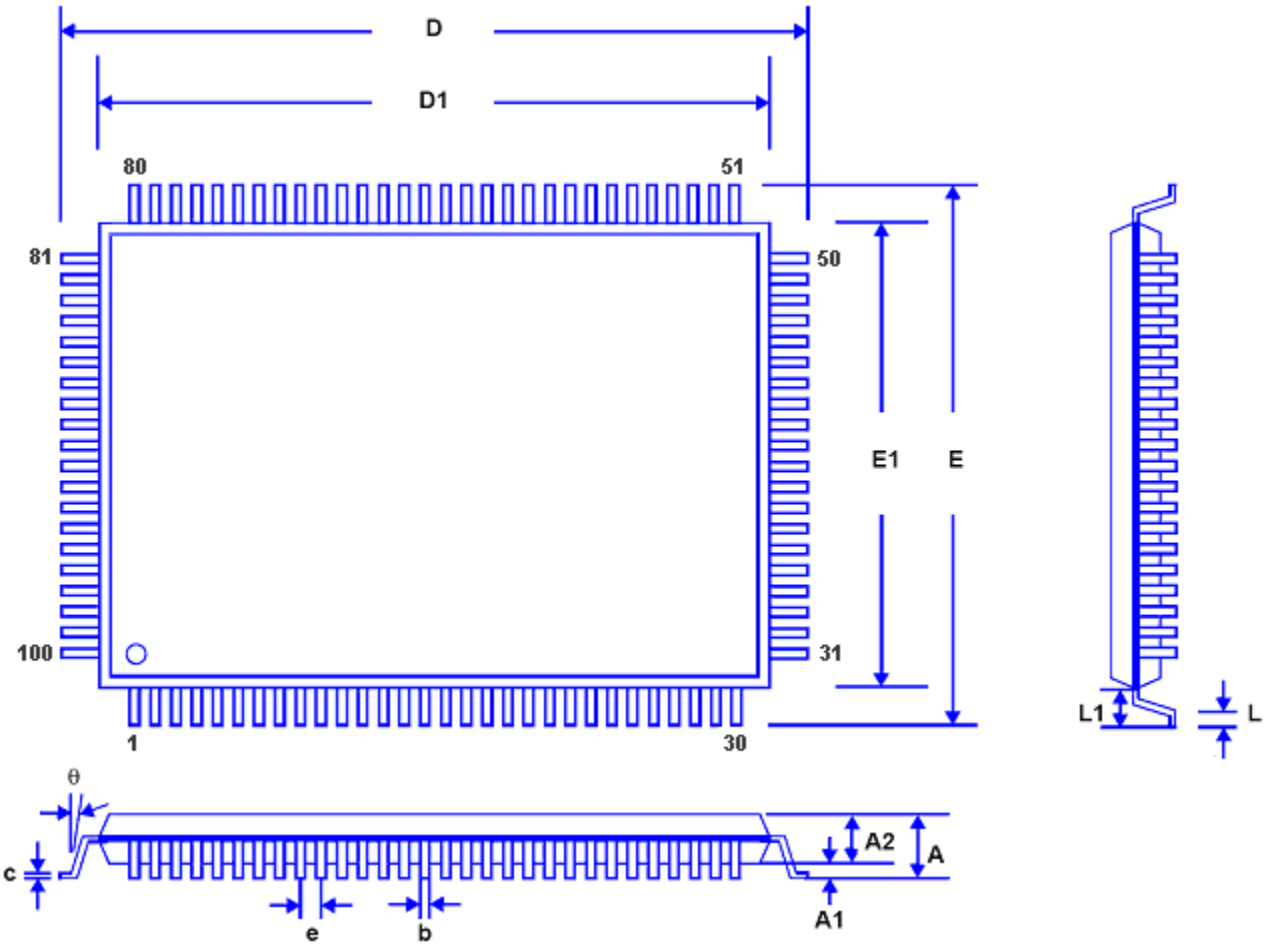
VDD = 5V, TA = -45^{o}C ~ 85^{o}C

SYMBOL	PARAMETER	MIN	TYP	MAX	UNIT	NOTE
Power Supply Current						
IDD (Normal)	Normal mode Supply Current using XOSC (16MHz) as clock source	-	30	40	mA	
IDD (PMM)	PMM mode Supply Current using XOSC (16MHz) as clock source	-	5.8	6.5	mA	
IDD (LPM)	Low Power mode Supply Current using RTC (32KHz) as clock source	-	3.7	4.2	mA	
IDD (Stop)	Stop mode Supply Current using XOSC (16MHz) as clock source	-	720	1000	uA	
I/O Characteristics						
V_{IH}	Input High Level	0.7VDD	-	VDD+0.5	V	
V_{IL}	Input Low Level	-0.5	-	0.25VDD	V	
V_{INHYS}	Input Hysteresis	0.1	0.2	0.3	V	
V_{OH}	Output High Level @ 4mA	VDD-0.6	-	VDD	V	
V_{OL}	Output Low Level @ 4mA	0	-	0.4	V	
$V_{IH,RSTN}$	Input High Level, RSTN	3.0		VDD	V	
$V_{IL,RSTN}$	Input Low Level, RSTN	0	-	1.5	V	
R_{PL}	Input Pull Up Resistor	50K	-	500K	Ohm	
Internal 2.5V Regulator						
V25	Internal 2.5V Regulator Output	2.375	2.5	2.625	V	

28. PACKAGE OUTLINE

28.1 Package Outline

100-pin QFP



Symbol	Dimensions in Millimeters		
	MIN	NOM	MAX
A	-	-	-
A1	0.10	-	0.50
A2	2.50	-	2.90
b	0.20	0.30	0.40
c	0.10	0.15	0.20
D	24.60	24.80	25.00
D1	19.90	20.00	20.10
e	0.498	0.65	0.802
E	18.60	18.80	19.00
E1	13.90	14.00	14.10
L	1.00	1.20	1.40
L1	-	2.40	-
θ	0	-	7
y	-	-	0.10

Notes:

- *JEDEC OUTLINE: MO-112 CC-1*
- *Datum plane H is located at the bottom of the mold parting line coincident with where the lead exists the body.*
- *Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25mm per side. Dimensions D1 and E1 do include mold mismatch and are determined at datum plane H.*
- *Dimension b does not include dambar protrusion.*

29. ORDERING INFORMATION

Premix	Part number	Green Package	Suffix
CS	8959	F: QFP	-I: Operating Temperature -40ºC ~ 85ºC

***Attachment**:*

Example 1:

```
            //Forward function (CAN1 can transmit the data received by CAN0 )
#include "reg52.h"
#include "absacc.h"
#define uchar unsigned char
#define uint  unsigned int
#define CAN0 0
#define CAN1 1
#define CAN0_ADDR    0X0B00
#define CAN1_ADDR   0X0D00

sbit P2_0 = P2 ^ 0;

sfr  EXIE = 0XE8;          //interrupt enable register
sbit EINT3 = EXIE ^ 2;  //CAN0 interrupt enable bit
sbit EINT4 = EXIE ^ 3;  //CAN1 interrupt enable bit

void write_reg(unsigned char can_no,unsigned int addr,unsigned char value)
{
   if(can_no)addr = CAN1_ADDR + addr;
   else addr = CAN0_ADDR + addr;
        XBYTE[addr]  =value;
}

uchar read_reg(unsigned char can_no,unsigned int addr)
{
   if(can_no)addr = CAN1_ADDR + addr;
   else addr = CAN0_ADDR + addr;
   return(XBYTE[addr]);
}

void can0_int3(void) interrupt 9                    //CAN0 receive interrupt subroutine
{
   EINT3 = 0;                                       // disable the CAN0 interrupt
   if(read_reg(CAN0,0X03)&0X01)                     //is the CAN0 receive interrupt?
        {
         write_reg(CAN0,1,0X40);                    //set the Forward Request bit to logic 1
        write_reg(CAN0,1,0X04);                     //release the receive buffer
        }
        EINT3 = 1;
}

void main(void)
{
   EINT3 = 1;    //enable CAN0 interrupt
   EA = 1;       //enable global interrupt
     XBYTE[0X0F52] = 0X0A;    //PIN75 and PIN76 are CAN0 RX and TX pin
                              //correspondingly, PIN18 and PIN19 are CAN1's RX
                              //and TX pin correspondingly
     write_reg(CAN0,0X09,0X01); //set bus-timing0 register
     write_reg(CAN0,0X0a,0X14); //set bus-timing1 register and the baud rate is 500kbps
                                //when the frequency of the crystal is 16mhz.
     write_reg(CAN1,0X09,0X00) ; //set bus-timing0 register
     write_reg(CAN1,0X0a,0X14); //set bus-timing1 register and the baud rate is 1mbps when
                                //the frequency of the crystal is 16mhz.
```

```
    write_reg(CAN0,0X04,0X01); //enable CAN0 bus error irq
    write_reg(CAN1,0X04,0X01) ;//enable CAN1 receive irq
    write_reg(CAN0,0X29,0XFf);  //set the filter 0 Mask Code bits are all logic 1
    write_reg(CAN0,0X2B,0XFf);
    write_reg(CAN0,0X2D,0XFf);
    write_reg(CAN0,0X2F,0XFf);
    write_reg(CAN0,0X05,0XFf);  //all the filters' forward function are enabled
    write_reg(CAN0,0X06,0XFf);
    write_reg(CAN0,0X07,0XFf);
    write_reg(CAN0,0X08,0XFf);
    write_reg(CAN0,0X00,0X00); //CAN0 enters into Operating mode
    write_reg(CAN1,0X00,0X00); //CAN1 enters into Operating mode
     while(1)
                {
                 P2_0 = ~P2_0;  //wait for receive interrupt
          };
}
```

Example 2:

```
                        //CAN0 dispatch
#include "reg52.h"
#include "absacc.h"
#define uchar unsigned char
#define uint  unsigned int
#define CAN0 0
#define CAN1 1
#define CAN0_ADDR    0X0B00
#define CAN1_ADDR   0X0D00
sbit P2_0 = P2 ^ 0;

void delay(uint a)
{
        while(a--);
}
void write_reg(unsigned char can_no,unsigned int addr,unsigned char value)
{
   if(can_no)addr = CAN1_ADDR + addr;
   else addr = CAN0_ADDR + addr;
        XBYTE[addr]  =value;
}
void main(void)
{
         XBYTE[0X0F52] = 0X0A;          //PIN75 and PIN76 are CAN0 RX and TX pin
                                       // correspondingly, PIN18 and PIN19 are CAN1's
                                       //RX and TX pin correspondingly
          write_reg(CAN0,0X09,0X01) ;//set bus-timing0 register
         write_reg(CAN0,0X0a,0X14); // set bus-timing1 register and the baud rate is 500kbps
                                       //when the frequency of the crystal is 16mhz.
          delay(10000);
         write_reg(CAN0,36,0X60);   // write-protect is closed
        write_reg(CAN0,39,0X01);     // set the dispatch time register
      write_reg(CAN0,0X00,0X00); // CAN0 enters into Operating
                                    //mode and start dispatching periodically
         while(1)
         {
            P2_0 = ~P2_0;  //waiting
          };
}
```

Example 3:

```
            //CAN0 sends a frame to CAN-bus
#include "reg52.h"
#include "absacc.h"
#define uchar unsigned char
#define uint  unsigned int
#define CAN0 0
#define CAN1 1
#define CAN0_ADDR    0X0B00
#define CAN1_ADDR   0X0D00
sbit P2_0 = P2 ^ 0;

void delay(uint a)
{
          while(a--);
}
void write_reg(unsigned char can_no,unsigned int addr,unsigned char value)
{
   if(can_no)addr = CAN1_ADDR + addr;
   else addr = CAN0_ADDR + addr;
          XBYTE[addr]  =value;
}
void main(void)
{
            XBYTE[0X0F52] = 0X0A;       //PIN57 and PIN58 are CAN1 RX and TX pin
                                        // correspondingly, PIN97 and PIN98 are CAN0's
                                        // RX and TX pin correspondingly
          write_reg(CAN0,0X09,0X01); //set bus-timing0 register
            write_reg(CAN0,0X0a,0X14);// set bus-timing1 register and the baud rate is 500kbps
                                      //when the frequency of the crystal is 16mhz.

          write_reg(CAN0,0X00,0X00)  ;// CAN0 enters into Operating mode
                                     // and start dispatching periodically
           write_reg(CAN0,0X17,0X88);  //write the transmit buffer
           write_reg(CAN0,0X18,0X11);
            write_reg(CAN0,0X19,0X22);
            write_reg(CAN0,0X1a,0X33);
            write_reg(CAN0,0X1b,0X44);
            write_reg(CAN0,0X1c,0X01);
            write_reg(CAN0,0X1d,0X02);
            write_reg(CAN0,0X1e,0X03);
            write_reg(CAN0,0X1f,0X04);
            write_reg(CAN0,0X20,0X05);
            write_reg(CAN0,0X21,0X06);
            write_reg(CAN0,0X22,0X07);
            write_reg(CAN0,0X23,0X08);
            while(1)
            {
                delay(10000);       // delays
                write_reg(CAN0,0X01,0X01);//set the transmit request bit to logic 1;
             };
}
```

CONFIDENTIALITY

DOCUMENT TEMPLATE:

Sheet Setting:	Up margin 2cm, Down margin 1 cm, Left margin 1.75cm, Right margin 1.75cm
Segment	To Be Define
Title fonts	Level one, Arial 14 + Bold Level two, Arial 14 Level three, Arial 14 Level four, Arial 12 + Underline
Content	Arial 10
Register Description	Register title, Arial 10 + Bold Shade of register title, To Be Define Content in table, Arial 9
Table	Table name, Arial 10 + Bold Table title, Arial 9 + Bold Content in Table, Arial 9
Diagram	Diagram Title, Arial 12 + Bold Diagram name, Arial 12 + italic
Note	Arial 10 + italic

REVISION HISTORY:

Revision	Date	Description	Prepare by
Rev 0.9	2009/09/23	First Engineering spec.	Tim Chung, Lawrence Lin
Rev 0.92	2015/11/19	Delete ADC function	Tim Chung