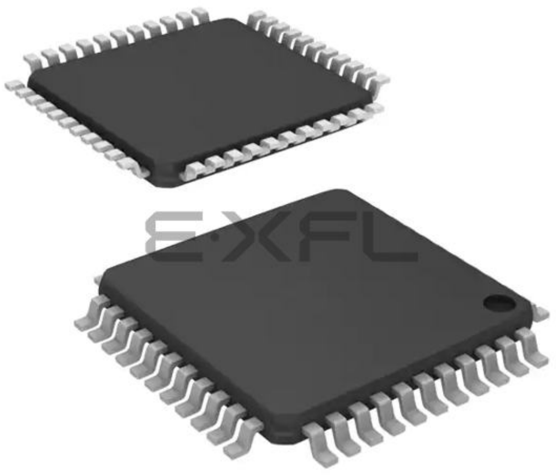
Microchip Technology - DSPIC33EP512GM604-I/PT Datasheet



Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	dsPIC
Core Size	16-Bit
Speed	70 MIPs
Connectivity	CANbus, I²C, IrDA, LINbus, QEI, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, I²S, Motor Control PWM, POR, PWM, WDT
Number of I/O	35
Program Memory Size	512KB (170K x 24)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	48K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	A/D 18x10b/12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-TQFP
Supplier Device Package	44-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/dspic33ep512gm604-i-pt

3.0 CPU

Note 1: This data sheet summarizes the features of the dsPIC33EPXXXGM3XX/6XX/7XX family of devices. It is not intended to be a comprehensive reference source. To complement the information in this data sheet, refer to the "*dsPIC33/PIC24 Family Reference Manual*", **"CPU"** (DS70359), which is available from the Microchip web site (www.microchip.com).

2: Some registers and associated bits described in this section may not be available on all devices. Refer to **Section 4.0 "Memory Organization"** in this data sheet for device-specific register and bit information.

The CPU has a 16-bit (data) modified Harvard architecture with an enhanced instruction set, including significant support for digital signal processing. The CPU has a 24-bit instruction word, with a variable length opcode field. The Program Counter (PC) is 23 bits wide and addresses up to 4M x 24 bits of user program memory space.

An instruction prefetch mechanism helps maintain throughput and provides predictable execution. Most instructions execute in a single-cycle, effective execution rate, with the exception of instructions that change the program flow, the double-word move (`MOV.D`) instruction, PSV accesses and the table instructions. Overhead-free program loop constructs are supported using the `DO` and `REPEAT` instructions, both of which are interruptible at any point.

3.1 Registers

The dsPIC33EPXXXGM3XX/6XX/7XX devices have sixteen 16-bit Working registers in the programmer's model. Each of the Working registers can act as a data, address or address offset register. The 16th Working register (W15) operates as a Software Stack Pointer for interrupts and calls.

3.2 Instruction Set

The device instruction set has two classes of instructions: the MCU class of instructions and the DSP class of instructions. These two instruction classes are seamlessly integrated into the architecture and execute from a single execution unit. The instruction set includes many addressing modes and was designed for optimum C compiler efficiency.

3.3 Data Space Addressing

The Base Data Space can be addressed as 4K words or 8 Kbytes and is split into two blocks, referred to as X and Y data memory. Each memory block has its own independent Address Generation Unit (AGU). The MCU class of instructions operate solely through the X memory AGU, which accesses the entire memory map as one linear Data Space. On dsPIC33EP devices, certain DSP instructions operate through the X and Y AGUs to support dual operand reads, which splits the data address space into two parts. The X and Y Data Space boundary is device-specific.

The upper 32 Kbytes of the Data Space memory map can optionally be mapped into Program Space at any 16K program word boundary. The program-to-Data Space mapping feature, known as Program Space Visibility (PSV), lets any instruction access Program Space as if it were Data Space. Moreover, the Base Data Space address is used in conjunction with a Data Space Read or Write Page register (DSRPAG or DSWPAG) to form an Extended Data Space (EDS) address. The EDS can be addressed as 8M words or 16 Mbytes. Refer to **"Data Memory"** (DS70595) and **"Program Memory"** (DS70613) in the *"dsPIC33/PIC24 Family Reference Manual"* for more details on EDS, PSV and table accesses.

On dsPIC33EP devices, overhead-free circular buffers (Modulo Addressing) are supported in both X and Y address spaces. The Modulo Addressing removes the software boundary checking overhead for DSP algorithms. The X AGU circular addressing can be used with any of the MCU class of instructions. The X AGU also supports Bit-Reversed Addressing to greatly simplify input or output data reordering for radix-2 FFT algorithms.

3.4 Addressing Modes

The CPU supports these addressing modes:

- Inherent (no operand)
- Relative
- Literal
- Memory Direct
- Register Direct
- Register Indirect

Each instruction is associated with a predefined addressing mode group, depending upon its functional requirements. As many as six addressing modes are supported for each instruction.

TABLE 4-42: CTMU REGISTER MAP

SFR Name	Addr.	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
CTMUCON1	033A	CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	CTTRIG	—	—	—	—	—	—	—	—	0000
CTMUCON2	033C	EDG1MOD	EDG1POL	EDG1SEL3	EDG1SEL2	EDG1SEL1	EDG1SEL0	EDG2STAT	EDG1STAT	EDG2MOD	EDG2POL	EDG2SEL3	EDG2SEL2	EDG2SEL1	EDG2SEL0	—	—	0000
CTMUICON	033E	ITRIM5	ITRIM4	ITRIM3	ITRIM2	ITRIM1	ITRIM0	IRNG1	IRNG0	—	—	—	—	—	—	—	—	0000

Legend: — = unimplemented, read as ‘0’. Reset values are shown in hexadecimal.

TABLE 4-43: JTAG INTERFACE REGISTER MAP

SFR Name	Addr.	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
JDATAH	0FF0	—	—	—	—	JDATAH<27:16>												xxxx
JDATAL	0FF2	JDATAL<15:0>																0000

Legend: x = unknown value on Reset; — = unimplemented, read as ‘0’. Reset values are shown in hexadecimal.

TABLE 4-44: REAL-TIME CLOCK AND CALENDAR REGISTER MAP

File Name	Addr.	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	All Resets
ALRMVAL	0620	Alarm Value Register Window Based on ALRMPTR<1:0>																xxxx
ALCFGRPT	0622	ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0	ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0	0000
RTCVAL	0624	RTCC Value Register Window Based on RTCPTR<1:0>																xxxx
RCFGCAL	0626	RTCEN	—	RTCWREN	RTCSYNC	HALFSEC	RTCOE	RTCPTR1	RTCPTR0	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0	0000

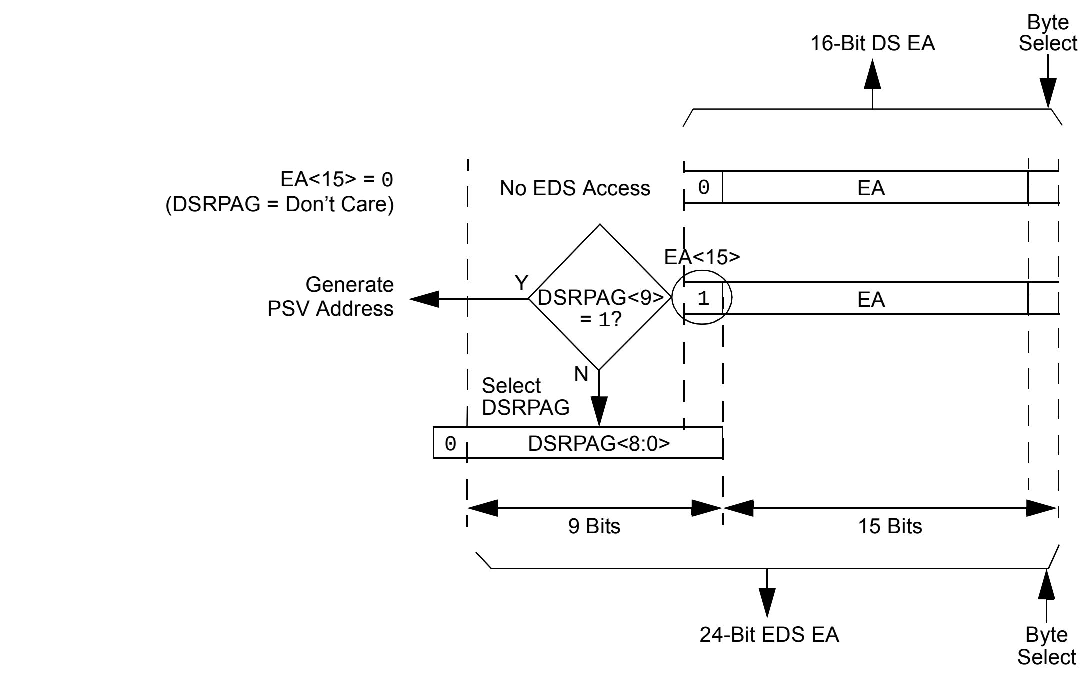
Legend: x = unknown value on Reset; — = unimplemented, read as ‘0’. Reset values are shown in hexadecimal.

4.3.1 PAGED MEMORY SCHEME

The dsPIC33EPXXXGM3XX/6XX/7XX architecture extends the available Data Space through a paging scheme, which allows the available Data Space to be accessed using `MOV` instructions in a linear fashion for pre- and post-modified Effective Addresses (EA). The upper half of the Base Data Space address is used in conjunction with the Data Space Page registers, the 10-bit Data Space Read Page register (DSRPAG) or the 9-bit Data Space Write Page register (DSWPAG), to form an Extended Data Space (EDS) address, or Program Space Visibility (PSV) address. The Data Space Page registers are located in the SFR space.

Construction of the EDS address is shown in Figure 4-8. When DSRPAG<9> = `0` and the base address bit, EA<15> = `1`, the DSRPAG<8:0> bits are concatenated onto EA<14:0> to form the 24-bit EDS read address. Similarly, when the base address bit, EA<15> =`1`, the DSWPAG<8:0> bits are concatenated onto EA<14:0> to form the 24-bit EDS write address.

FIGURE 4-8: EXTENDED DATA SPACE (EDS) READ ADDRESS GENERATION



Note: DS read access when DSRPAG = 0x000 will force an address error trap.

4.5.3 MODULO ADDRESSING APPLICABILITY

Modulo Addressing can be applied to the Effective Address (EA) calculation associated with any W register. Address boundaries check for addresses equal to:

- The upper boundary addresses for incrementing buffers
- The lower boundary addresses for decrementing buffers

It is important to realize that the address boundaries check for addresses less than or greater than the upper (for incrementing buffers) and lower (for decrementing buffers) boundary addresses (not just equal to). Address changes can, therefore, jump beyond boundaries and still be adjusted correctly.

Note: The modulo corrected Effective Address is written back to the register only when Pre-Modify or Post-Modify Addressing mode is used to compute the Effective Address. When an address offset (such as [W7 + W2]) is used, Modulo Addressing correction is performed, but the contents of the register remain unchanged.

4.6 Bit-Reversed Addressing

Bit-Reversed Addressing mode is intended to simplify data reordering for radix-2 FFT algorithms; it is supported by the X AGU for data writes only.

The modifier, which can be a constant value or register contents, is regarded as having its bit order reversed. The address source and destination are kept in normal order. Thus, the only operand requiring reversal is the modifier.

4.6.1 BIT-REVERSED ADDRESSING IMPLEMENTATION

Bit-Reversed Addressing mode is enabled when all of these conditions are met:

- BWM bits (W register selection) in the MODCON register are any value other than ‘`1111`’ (the stack cannot be accessed using Bit-Reversed Addressing)
- The BREN bit is set in the XBREV register
- The addressing mode used is Register Indirect with Pre-Increment or Post-Increment

If the length of a bit-reversed buffer is $M = 2^N$ bytes, the last ‘N’ bits of the data buffer start address must be zeros.

XB<14:0> is the Bit-Reversed Addressing modifier, or ‘pivot point’, which is typically a constant. In the case of an FFT computation, its value is equal to half of the FFT data buffer size.

Note: All bit-reversed EA calculations assume word-sized data (LSb of every EA is always clear). The XB value is scaled accordingly to generate compatible (byte) addresses.

When enabled, Bit-Reversed Addressing is executed only for Register Indirect with Pre-Increment or Post-Increment Addressing and word-sized data writes. It does not function for any other addressing mode or for byte-sized data and normal addresses are generated instead. When Bit-Reversed Addressing is active, the W Address Pointer is always added to the address modifier (XB) and the offset associated with the Register Indirect Addressing mode is ignored. In addition, as word-sized data is a requirement, the LSb of the EA is ignored (and always clear).

Note: Modulo Addressing and Bit-Reversed Addressing can be enabled simultaneously using the same W register, but Bit-Reversed Addressing operation will always take precedence for data writes when enabled.

If Bit-Reversed Addressing has already been enabled by setting the BREN (XBREV<15>) bit, a write to the XBREV register should not be immediately followed by an indirect read operation using the W register that has been designated as the Bit-Reversed Pointer.

4.7.1 DATA ACCESS FROM PROGRAM MEMORY USING TABLE INSTRUCTIONS

The `TBLRDL` and `TBLWTL` instructions offer a direct method of reading or writing the lower word of any address within the Program Space without going through Data Space. The `TBLRDH` and `TBLWTH` instructions are the only method to read or write the upper 8 bits of a Program Space word as data.

The PC is incremented by two for each successive 24-bit program word. This allows program memory addresses to directly map to Data Space addresses. Program memory can thus be regarded as two 16-bit-wide word address spaces, residing side by side, each with the same address range. `TBLRDL` and `TBLWTL` access the space that contains the least significant data word. `TBLRDH` and `TBLWTH` access the space that contains the upper data byte.

Two table instructions are provided to move byte or word-sized (16-bit) data to and from Program Space. Both function as either byte or word operations.

- `TBLRDL` (Table Read Low):
  - In Word mode, this instruction maps the lower word of the Program Space location (P<15:0>) to a data address (D<15:0>)
  - In Byte mode, either the upper or lower byte of the lower program word is mapped to the lower byte of a data address. The upper byte is selected when Byte Select is ‘`1`’; the lower byte is selected when it is ‘`0`’.
- `TBLRDH` (Table Read High):
  - In Word mode, this instruction maps the entire upper word of a program address (P<23:16>) to a data address. The ‘phantom’ byte (D<15:8>) is always ‘`0`’.
  - In Byte mode, this instruction maps the upper or lower byte of the program word to D<7:0> of the data address in the `TBLRDL` instruction. The data is always ‘`0`’ when the upper ‘phantom’ byte is selected (Byte Select = `1`).

In a similar fashion, two table instructions, `TBLWTH` and `TBLWTL`, are used to write individual bytes or words to a Program Space address. The details of their operation are explained in **Section 5.0 “Flash Program Memory”**.

For all table operations, the area of program memory space to be accessed is determined by the Table Page register (TBLPAG). TBLPAG covers the entire program memory space of the device, including user application and configuration spaces. When TBLPAG<7> = `0`, the table page is located in the user memory space. When TBLPAG<7> = `1`, the page is located in configuration space.

FIGURE 4-17: ACCESSING PROGRAM MEMORY WITH TABLE INSTRUCTIONS

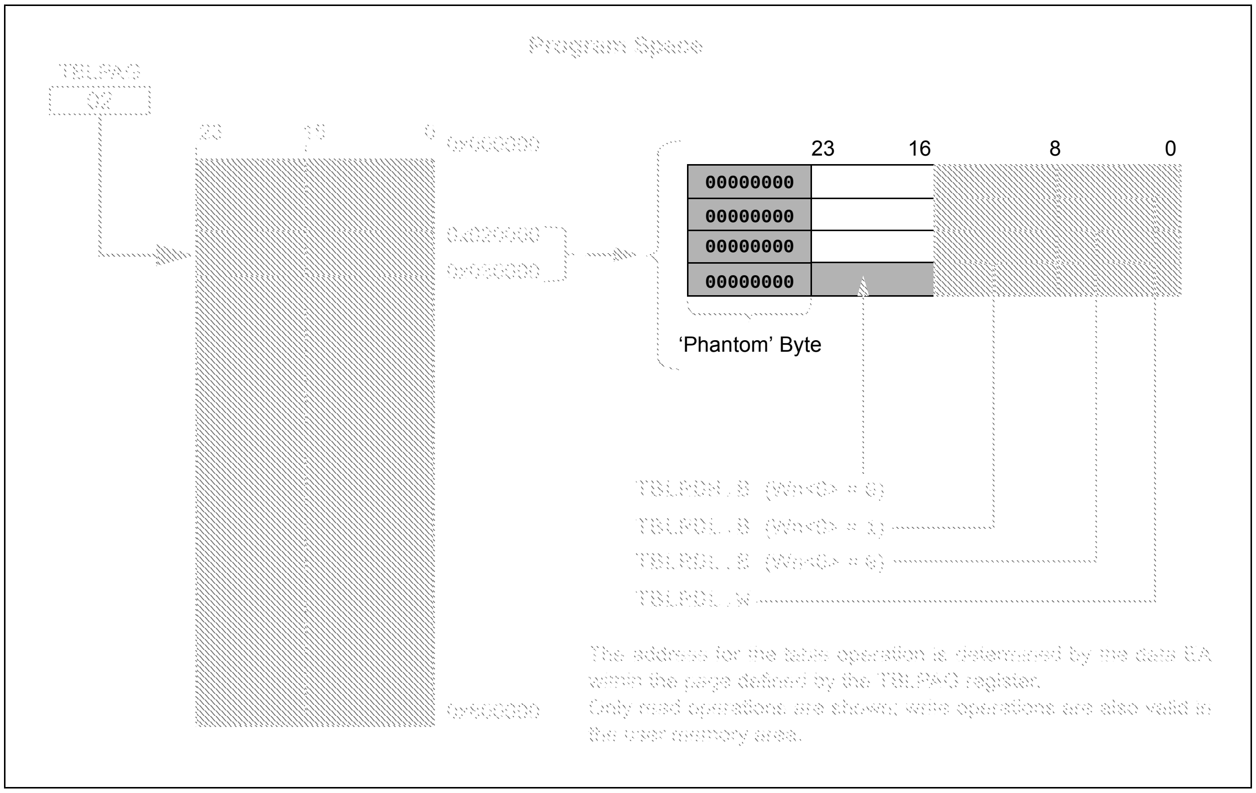

Program Space
TBLPAG
02
23
15
0
0x000000
0x020000
0x030000
0x800000
23
16
8
0
00000000
00000000
00000000
00000000
‘Phantom’ Byte
TBLRDH.B (Wn<0> = 0)
TBLRDL.B (Wn<0> = 1)
TBLRDL.B (Wn<0> = 0)
TBLRDL.W
The address for the table operation is determined by the data EA within the page defined by the TBLPAG register.
Only read operations are shown; write operations are also valid in the user memory area.

TABLE 9-1: CONFIGURATION BIT VALUES FOR CLOCK SELECTION

Oscillator Mode	Oscillator Source	POSCMD<1:0>	FNOSC<2:0>	See Notes
Fast RC Oscillator with Divide-by-N (FRCDIVN)	Internal	xx	111	**1, 2**
Fast RC Oscillator with Divide-by-16 (FRCDIV16)	Internal	xx	110	**1**
Low-Power RC Oscillator (LPRC)	Internal	xx	101	**1**
Secondary (Timer1) Oscillator (SOSC)	Secondary	xx	100	**1**
Primary Oscillator (HS) with PLL (HSPLL)	Primary	10	011	
Primary Oscillator (XT) with PLL (XTPLL)	Primary	01	011	
Primary Oscillator (EC) with PLL (ECPLL)	Primary	00	011	**1**
Primary Oscillator (HS)	Primary	10	010	
Primary Oscillator (XT)	Primary	01	010	
Primary Oscillator (EC)	Primary	00	010	**1**
Fast RC Oscillator (FRC) with Divide-by-N and PLL (FRCPLL)	Internal	xx	001	**1**
Fast RC Oscillator (FRC)	Internal	xx	000	**1**

Note 1: OSC2 pin function is determined by the OSCIOFNC Configuration bit.

2: This is the default oscillator mode for an unprogrammed (erased) device.

11.4 Peripheral Pin Select (PPS)

A major challenge in general purpose devices is providing the largest possible set of peripheral features while minimizing the conflict of features on I/O pins. The challenge is even greater on low pin count devices. In an application where more than one peripheral needs to be assigned to a single pin, inconvenient work arounds in application code or a complete redesign may be the only option.

Peripheral Pin Select configuration provides an alternative to these choices by enabling peripheral set selection and their placement on a wide range of I/O pins. By increasing the pinout options available on a particular device, users can better tailor the device to their entire application, rather than trimming the application to fit the device.

The Peripheral Pin Select configuration feature operates over a fixed subset of digital I/O pins. Users may independently map the input and/or output of most digital peripherals to any one of these I/O pins. Hardware safeguards are included that prevent accidental or spurious changes to the peripheral mapping once it has been established.

11.4.1 AVAILABLE PINS

The number of available pins is dependent on the particular device and its pin count. Pins that support the Peripheral Pin Select feature include the designation, “RPn” or “RPIn”, in their full pin designation, where “n” is the remappable pin number. “RP” is used to designate pins that support both remappable input and output functions, while “RPI” indicates pins that support remappable input functions only.

11.4.2 AVAILABLE PERIPHERALS

The peripherals managed by the Peripheral Pin Select are all digital only peripherals. These include general serial communications (UART and SPI), general purpose timer clock inputs, timer-related peripherals (input capture and output compare) and interrupt-on-change inputs.

In comparison, some digital only peripheral modules are never included in the Peripheral Pin Select feature. This is because the peripheral’s function requires special I/O circuitry on a specific port and cannot be easily connected to multiple pins. These modules include I^2C™ and the PWM. A similar requirement excludes all modules with analog inputs, such as the A/D Converter.

A key difference between remappable and non-remappable peripherals is that remappable peripherals are not associated with a default I/O pin. The peripheral must always be assigned to a specific I/O pin before it can be used. In contrast, non-remappable peripherals are always available on a default pin, assuming that the peripheral is active and not conflicting with another peripheral.

When a remappable peripheral is active on a given I/O pin, it takes priority over all other digital I/O and digital communication peripherals associated with the pin. Priority is given regardless of the type of peripheral that is mapped. Remappable peripherals never take priority over any analog functions associated with the pin.

11.4.3 CONTROLLING PERIPHERAL PIN SELECT

Peripheral Pin Select features are controlled through two sets of SFRs: one to map peripheral inputs and one to map outputs. Because they are separately controlled, a particular peripheral’s input and output (if the peripheral has both) can be placed on any selectable function pin without constraint.

The association of a peripheral to a peripheral-selectable pin is handled in two different ways, depending on whether an input or output is being mapped.

6. The Peripheral Pin Select (PPS) pin mapping rules are as follows:
    a) Only one "output" function can be active on a given pin at any time, regardless if it is a dedicated or remappable function (one pin, one output).
    b) It is possible to assign a "remappable output" function to multiple pins and externally short or tie them together for increased current drive.
    c) If any "dedicated output" function is enabled on a pin, it will take precedence over any remappable "output" function.
    d) If any "dedicated digital" (input or output) function is enabled on a pin, any number of "input" remappable functions can be mapped to the same pin.
    e) If any "dedicated analog" function(s) are enabled on a given pin, "digital input(s)" of any kind will all be disabled, although a single "digital output", at the user's cautionary discretion, can be enabled and active as long as there is no signal contention with an external analog input signal. For example, it is possible for the ADCx to convert the digital output logic level or to toggle a digital output on a comparator or ADCx input provided there is no external analog input, such as for a built-in self-test.
    f) Any number of "input" remappable functions can be mapped to the same pin(s) at the same time, including to any pin with a single output from either a dedicated or remappable "output".
    g) The TRIS registers control *only* the digital I/O output buffer. Any other dedicated or remappable active "output" will automatically override the TRIS setting. The TRIS register *does not* control the digital logic "input" buffer. Remappable digital "inputs" do not automatically override TRIS settings, which means that the TRIS bit must be set to input for pins with only remappable input function(s) assigned.
    h) All analog pins are enabled by default after any Reset and the corresponding digital input buffer on the pin is disabled. Only the Analog Pin Select registers control the digital input buffer, *not* the TRIS register. The user must disable the analog function on a pin using the Analog Pin Select registers in order to use any "digital input(s)" on a corresponding pin, no exceptions.

NOTES:

REGISTER 17-17: INTxTMRH: INTERVAL TIMERx HIGH WORD REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INTTMR<31:24>							
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INTTMR<23:16>							
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 15-0 **INTTMR<31:16>:** High Word Used to Form 32-Bit Interval Timerx Register (INTxTMR) bits

REGISTER 17-18: INTxTMRL: INTERVAL TIMERx LOW WORD REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INTTMR<15:8>							
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INTTMR<7:0>							
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 15-0 **INTTMR<15:0>:** Low Word Used to Form 32-Bit Interval Timerx Register (INTxTMR) bits

REGISTER 21-6: CxINTF: CANx INTERRUPT FLAG REGISTER

U-0	U-0	R-0	R-0	R-0	R-0	R-0	R-0
—	—	TXBO	TXBP	RXBP	TXWAR	RXWAR	EWARN
bit 15							bit 8

R/C-0	R/C-0	R/C-0	U-0	R/C-0	R/C-0	R/C-0	R/C-0
IVRIF	WAKIF	ERRIF	—	FIFOIF	RBOVIF	RBIF	TBIF
bit 7							bit 0

Legend:	C = Writable bit, but only ‘0’ can be written to clear the bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 15-14	**Unimplemented:** Read as ‘0’
bit 13	**TXBO:** Transmitter in Error State Bus Off bit 1 = Transmitter is in Bus Off state 0 = Transmitter is not in Bus Off state
bit 12	**TXBP:** Transmitter in Error State Bus Passive bit 1 = Transmitter is in Bus Passive state 0 = Transmitter is not in Bus Passive state
bit 11	**RXBP:** Receiver in Error State Bus Passive bit 1 = Receiver is in Bus Passive state 0 = Receiver is not in Bus Passive state
bit 10	**TXWAR:** Transmitter in Error State Warning bit 1 = Transmitter is in Error Warning state 0 = Transmitter is not in Error Warning state
bit 9	**RXWAR:** Receiver in Error State Warning bit 1 = Receiver is in Error Warning state 0 = Receiver is not in Error Warning state
bit 8	**EWARN:** Transmitter or Receiver in Error State Warning bit 1 = Transmitter or receiver is in Error Warning state 0 = Transmitter or receiver is not in Error Warning state
bit 7	**IVRIF:** Invalid Message Interrupt Flag bit 1 = Interrupt request has occurred 0 = Interrupt request has not occurred
bit 6	**WAKIF:** Bus Wake-up Activity Interrupt Flag bit 1 = Interrupt request has occurred 0 = Interrupt request has not occurred
bit 5	**ERRIF:** Error Interrupt Flag bit (multiple sources in CxINTF<13:8> register) 1 = Interrupt request has occurred 0 = Interrupt request has not occurred
bit 4	**Unimplemented:** Read as ‘0’
bit 3	**FIFOIF:** FIFO Almost Full Interrupt Flag bit 1 = Interrupt request has occurred 0 = Interrupt request has not occurred
bit 2	**RBOVIF:** RX Buffer Overflow Interrupt Flag bit 1 = Interrupt request has occurred 0 = Interrupt request has not occurred

REGISTER 21-26: CxTRmnCON: CANx TX/RX BUFFER mn CONTROL REGISTER (m = 0,2,4,6; n = 1,3,5,7)

R/W-0	R-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
TXENn	TXABTn	TXLARBn	TXERRn	TXREQn	RTRENn	TXnPRI1	TXnPRI0
bit 15							bit 8

R/W-0	R-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
TXENm	TXABTm(1)	TXLARBm(1)	TXERRm(1)	TXREQm	RTRENm	TXmPRI1	TXmPRI0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 15-8 See Definition for bits 7-0, controls Buffer n

bit 7 **TXENm:** TX/RX Buffer Selection bit
1 = Buffer, TRBn, is a transmit buffer
0 = Buffer, TRBn, is a receive buffer

bit 6 **TXABTm:** Message Aborted bit(1)
1 = Message was aborted
0 = Message completed transmission successfully

bit 5 **TXLARBm:** Message Lost Arbitration bit(1)
1 = Message lost arbitration while being sent
0 = Message did not lose arbitration while being sent

bit 4 **TXERRm:** Error Detected During Transmission bit(1)
1 = A bus error occurred while the message was being sent
0 = A bus error did not occur while the message was being sent

bit 3 **TXREQm:** Message Send Request bit
1 = Requests that a message be sent; the bit automatically clears when the message is successfully sent
0 = Clearing the bit to ‘0’ while set requests a message abort

bit 2 **RTRENm:** Auto-Remote Transmit Enable bit
1 = When a remote transmit is received, TXREQx will be set
0 = When a remote transmit is received, TXREQx will be unaffected

bit 1-0 **TXmPRI<1:0>:** Message Transmission Priority bits
11 = Highest message priority
10 = High intermediate message priority
01 = Low intermediate message priority
00 = Lowest message priority

Note 1: This bit is cleared when TXREQx is set.

Note: The buffers, SIDx, EIDx, DLCx, Data Field, and Receive Status registers, are located in DMA RAM.

REGISTER 25-3: PTGBTE: PTG BROADCAST TRIGGER ENABLE REGISTER(1,2) (CONTINUED)

bit 4	**OC1CS:** Clock Source for OC1 bit 1 = Generates clock pulse when the broadcast command is executed 0 = Does not generate clock pulse when the broadcast command is executed
bit 3	**OC4TSS:** Trigger/Synchronization Source for OC4 bit 1 = Generates trigger/synchronization when the broadcast command is executed 0 = Does not generate trigger/synchronization when the broadcast command is executed
bit 2	**OC3TSS:** Trigger/Synchronization Source for OC3 bit 1 = Generates trigger/synchronization when the broadcast command is executed 0 = Does not generate trigger/synchronization when the broadcast command is executed
bit 1	**OC2TSS:** Trigger/Synchronization Source for OC2 bit 1 = Generates trigger/synchronization when the broadcast command is executed 0 = Does not generate trigger/synchronization when the broadcast command is executed
bit 0	**OC1TSS:** Trigger/Synchronization Source for OC1 bit 1 = Generates trigger/synchronization when the broadcast command is executed 0 = Does not generate trigger/synchronization when the broadcast command is executed

Note 1: This register is read-only when the PTG module is executing Step commands (PTGEN = 1 and PTGSTRT = 1).

2: This register is only used with the `PTGCTRL OPTION` = 1111 Step command.

FIGURE 26-3: USER-PROGRAMMABLE BLANKING FUNCTION BLOCK DIAGRAM

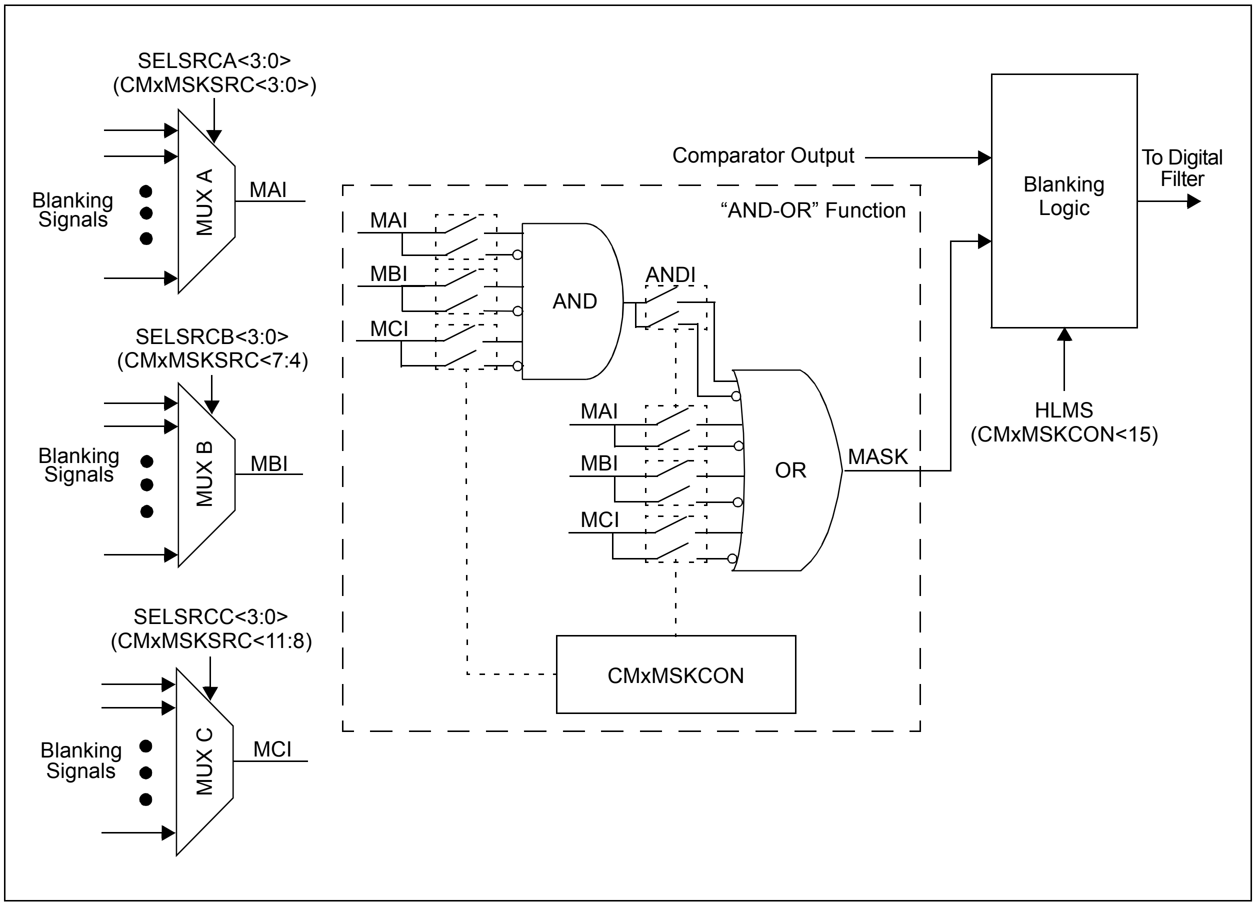
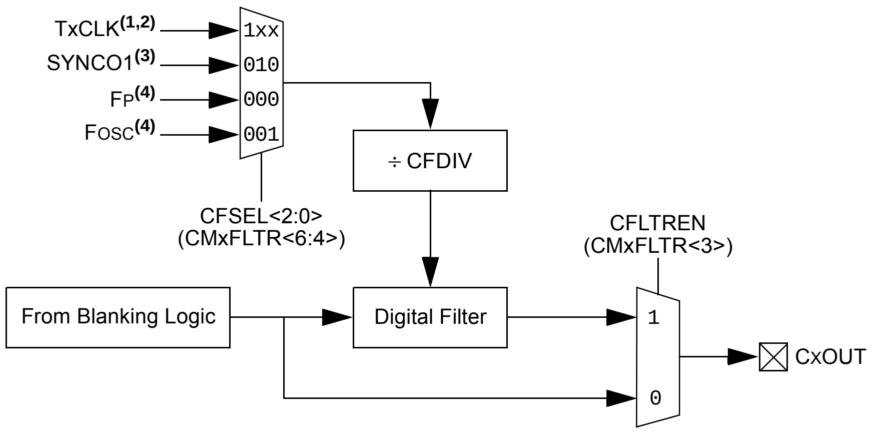


FIGURE 26-4: DIGITAL FILTER INTERCONNECT BLOCK DIAGRAM



Note 1: See the Type C Timer Block Diagram (Figure 13-2).
2: See the Type B Timer Block Diagram (Figure 13-1).
3: See the PWMx Module Register Interconnect Diagram (Figure 16-2).
4: See the Oscillator System Diagram (Figure 9-1).

REGISTER 26-2: CMxCON: OP AMP/COMPARATOR x CONTROL REGISTER (x = 1, 2, 3 OR 5) (CONTINUED)

bit 7-6	**EVPOL<1:0>:** Trigger/Event/Interrupt Polarity Select bits[3] 11 = Trigger/event/interrupt generated on any change of the comparator output (while CEVT = 0) 10 = Trigger/event/interrupt generated only on high-to-low transition of the polarity selected comparator output (while CEVT = 0) If CPOL = 1 (inverted polarity): Low-to-high transition of the comparator output. If CPOL = 0 (non-inverted polarity): High-to-low transition of the comparator output. 01 = Trigger/event/interrupt generated only on low-to-high transition of the polarity selected comparator output (while CEVT = 0) If CPOL = 1 (inverted polarity): High-to-low transition of the comparator output. If CPOL = 0 (non-inverted polarity): Low-to-high transition of the comparator output. 00 = Trigger/event/interrupt generation is disabled.
bit 5	**Unimplemented:** Read as '0'
bit 4	**CREF:** Comparator Reference Select bit (VIN+ input)[1] 1 = VIN+ input connects to internal CVREFIN voltage 0 = VIN+ input connects to CxIN1+ pin
bit 3-2	**Unimplemented:** Read as '0'
bit 1-0	**CCH<1:0>:** Op Amp/Comparator Channel Select bits[1] 11 = Inverting input of op amp/comparator connects to CxIN4- pin 10 = Inverting input of op amp/comparator connects to CxIN3- pin 01 = Inverting input of op amp/comparator connects to CxIN2- pin 00 = Inverting input of op amp/comparator connects to CxIN1- pin

Note 1: Inputs that are selected and not available will be tied to VSS. See the **"Pin Diagrams"** section for available inputs for each package.

2: The op amp and the comparator can be used simultaneously in these devices. The OPMODE bit only enables the op amp while the comparator is still functional.

3: After configuring the comparator, either for a high-to-low or low-to-high COUT transition (EVPOL<1:0> (CMxCON<7:6>) = 10 or 01), the Comparator Event bit, CEVT (CMxCON<9>), and the Comparator Combined Interrupt Flag, CMPIF (IFS1<2>), **must be cleared** before enabling the Comparator Interrupt Enable bit, CMPIE (IEC1<2>).

30.6 JTAG Interface

dsPIC33EPXXXGM3XX/6XX/7XX devices implement a JTAG interface, which supports boundary scan device testing. Detailed information on this interface is provided in future revisions of the document.

Note:	Refer to the *“dsPIC33/PIC24 Family Reference Manual”*, **“Programming and Diagnostics”** (DS70608) for further information on usage, configuration and operation of the JTAG interface.

30.7 In-Circuit Serial Programming

The dsPIC33EPXXXGM3XX/6XX/7XX devices can be serially programmed while in the end application circuit. This is done with two lines for clock and data, and three other lines for power, ground and the programming sequence. Serial programming allows customers to manufacture boards with unprogrammed devices and then program the device just before shipping the product. Serial programming also allows the most recent firmware or a custom firmware to be programmed. Refer to the *“dsPIC33E/PIC24E Flash Programming Specification for Devices with Volatile Configuration Bits”* (DS70663) for details about In-Circuit Serial Programming (ICSP).

Any of the three pairs of programming clock/data pins can be used:

- PGEC1 and PGED1
- PGEC2 and PGED2
- PGEC3 and PGED3

30.8 In-Circuit Debugger

When MPLAB® ICD 3 or the REAL ICE™ in-circuit emulator is selected as a debugger, the in-circuit debugging functionality is enabled. This function allows simple debugging functions when used with MPLAB X IDE. Debugging functionality is controlled through the PGECx (Emulation/Debug Clock) and PGEDx (Emulation/Debug Data) pin functions.

Any of the three pairs of debugging clock/data pins can be used:

- PGEC1 and PGED1
- PGEC2 and PGED2
- PGEC3 and PGED3

To use the in-circuit debugger function of the device, the design must implement ICSP connections to $\overline{\text{MCLR}}$, V_{DD}, V_{SS} and the PGECx/PGEDx pin pair. In addition, when the feature is enabled, some of the resources are not available for general use. These resources include the first 80 bytes of data RAM and two I/O pins (PGECx and PGEDx).

30.9 Code Protection and CodeGuard™ Security

The dsPIC33EPXXXGM3XX/6XX/7XX devices offer basic implementation of CodeGuard Security that supports only General Segment (GS) security. This feature helps protect individual Intellectual Property.

Note:	Refer to the *“dsPIC33/PIC24 Family Reference Manual”*, **“CodeGuard™ Security”** (DS70634) for further information on usage, configuration and operation of CodeGuard Security.

32.2 MPLAB XC Compilers

The MPLAB XC Compilers are complete ANSI C compilers for all of Microchip's 8, 16 and 32-bit MCU and DSC devices. These compilers provide powerful integration capabilities, superior code optimization and ease of use. MPLAB XC Compilers run on Windows, Linux or MAC OS X.

For easy source level debugging, the compilers provide debug information that is optimized to the MPLAB X IDE.

The free MPLAB XC Compiler editions support all devices and commands, with no time or memory restrictions, and offer sufficient code optimization for most applications.

MPLAB XC Compilers include an assembler, linker and utilities. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. MPLAB XC Compiler uses the assembler to produce its object file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command-line interface
- Rich directive set
- Flexible macro language
- MPLAB X IDE compatibility

32.3 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for PIC10/12/16/18 MCUs.

The MPASM Assembler generates relocatable object files for the MPLINK Object Linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, absolute LST files that contain source lines and generated machine code, and COFF files for debugging.

The MPASM Assembler features include:

- Integration into MPLAB X IDE projects
- User-defined macros to streamline assembly code
- Conditional assembly for multipurpose source files
- Directives that allow complete control over the assembly process

32.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK Object Linker combines relocatable objects created by the MPASM Assembler. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB Object Librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/library features include:

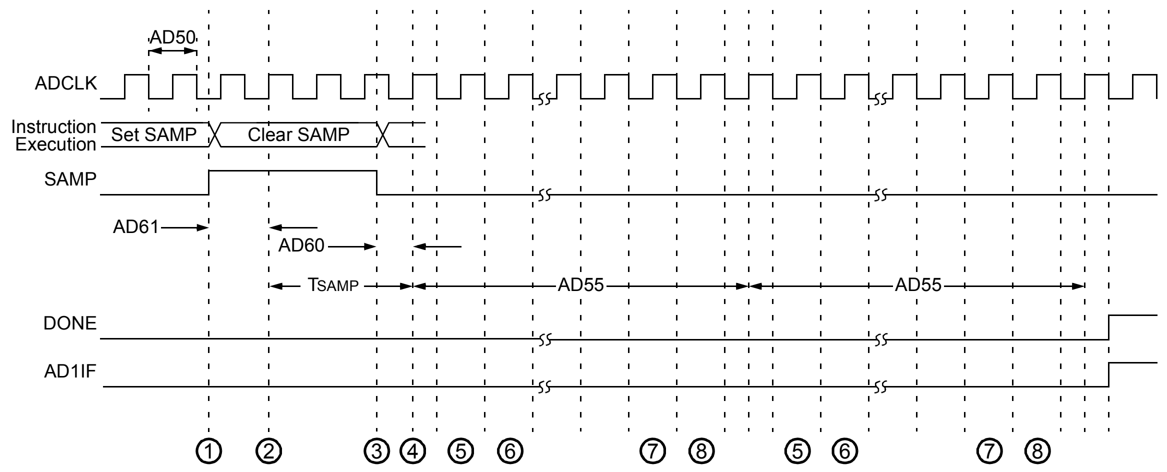
- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

32.5 MPLAB Assembler, Linker and Librarian for Various Device Families

MPLAB Assembler produces relocatable machine code from symbolic assembly language for PIC24, PIC32 and dsPIC DSC devices. MPLAB XC Compiler uses the assembler to produce its object file. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command-line interface
- Rich directive set
- Flexible macro language
- MPLAB X IDE compatibility

FIGURE 33-39: ADC1 CONVERSION (10-BIT MODE) TIMING CHARACTERISTICS (CHPS<1:0> = 01, SIMSAM = 0, ASAM = 0, SSRC<2:0> = 000, SSRCG = 0)



① – Software sets AD1CON1. SAMP to start sampling.

② – Sampling starts after discharge period. TSAMP is described in **"Analog-to-Digital Converter (ADC)"** (DS70621) of the *"dsPIC33/PIC24 Family Reference Manual"*.

③ – Software clears AD1CON1. SAMP to start conversion.

④ – Sampling ends, conversion sequence starts.

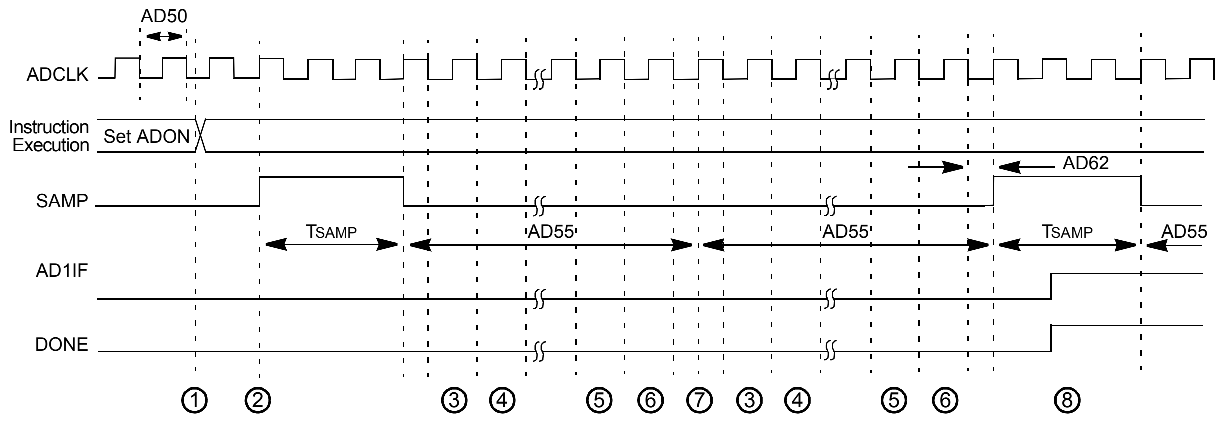
⑤ – Convert bit 9.

⑥ – Convert bit 8.

⑦ – Convert bit 0.

⑧ – One TAD for end of conversion.

FIGURE 33-40: ADC1 CONVERSION (10-BIT MODE) TIMING CHARACTERISTICS (CHPS<1:0> = 01, SIMSAM = 0, ASAM = 1, SSRC<2:0> = 111, SSRCG = 0, SAMC<4:0> = 00010)



① – Software sets AD1CON1. ADON to start ADC operation.

② – Sampling starts after discharge period. TSAMP is described in **"Analog-to-Digital Converter (ADC)"** (DS70621) of the *"dsPIC33/PIC24 Family Reference Manual"*.

③ – Convert bit 9.

④ – Convert bit 8.

⑤ – Convert bit 0.

⑥ – One TAD for end of conversion.

⑦ – Begin conversion of next channel.

⑧ – Sample for time specified by SAMC<4:0>.

NOTES: