TMS320C203 Revision 2.0 Device Update

Specific changes between 'C203 Rev 1.0 and Rev 2.0 devices

1. HOLD logic in MODE 0 is implemented in software logic.

2. Changes in software wait state values for write cycles. These are valid only for TMP devices.

3. TDDR bits in timer control register (TCR) can be both written to and read.

4. The FE (bit 10) and OE (bit 9) bits in I/O status register are cleared by writing 1 to them.

5. The URST bit (bit 13) in ASPCR resets the UART if set to 0, and enables the UART if set to 1.

6. Auto baud detection logic is fully functional.

7. The IN1 bit in SSPCR has been replaced by OVF bit to indicate overflow conditions in receive FIFO. It is cleared when the FIFO is read.

8. In all TMX and TMP devices, the SSP, UART, and DELTA interrupts should be cleared inside the interrupt service routine just before the return (RET). If this doesn't happen, a double interrupt will occur.

9. The TEST pin (pin1) should be connected to ground during normal operation of the 'C203. It is a no connection (NC) in the current documentation.

A. HOLD OPERATION

The TMS320C203 supports direct memory access (DMA) to its local (off-chip) program, data, and I/O spaces. Two signals, $\overline{HOLD}$, an input to the 'C203 and $\overline{HOLDA}$, an output, control this mechanism. If $\overline{HOLD}$ is asserted, the CPU along with user software logic can tri-state the address, data, and memory control signals ($\overline{PS}, \overline{DS}, \overline{IS}, \overline{STRB}, R/\overline{W}, \overline{RD}, \overline{WE}$). A $\overline{HOLDA}$ signal is issued acknowledging that the processor is relinquishing control of its external bus. Since the $\overline{HOLD}$ is shared with the $\overline{INT1}$ interrupt, the $\overline{HOLD}$ logic can be implemented if the MODE bit is zero in the interrupt control (IC at FFECh) register. At reset the MODE bit is set to zero, enabling HOLD logic. In MODE 0, execution of the IDLE instruction upon an INT1 interrupt will issue a HOLDA signal and tristate the busses. This feature with the IDLE instruction is only valid in MODE 0. It is important that the IDLE instruction be used with caution in MODE 0 to avoid bus control problems.

A.1 Hardware and software features used for HOLD function

1. In MODE 0 the INT1 interrupt is both negative and positive edge sensitive. This double edge interrupt sensing is useful in asserting and deasserting HOLDA.

2. Following a negative edge on the INT1 pin, if the MODE is zero and the interrupt INT1 is enabled, the CPU will branch to INT1 vector address (0x0002).

3. If the HOLD function is desired the user interrupt service routine should use an IDLE instruction with proper context saving to implement it. The assembly code below explains the method by which HOLDA is issued and the busses are put into tristate. The IDLE instruction should be placed inside the interrupt service program code to issue HOLDA.

INT1 Interrupt service routine for asserting HOLDA - an example

```
                .mmregs             ; include c2xx memory map registers
 IC             .set    0FFECh      ; Interrupt control register in I/O space
 ICSHDW         .set    060h        ; scratch pad location
*  Interrupt vectors
 reset          B       main        ; 0 - reset , Branch to main program on reset
 Int1h          B       Int1_hold   ; 1 - external interrupt 1 or HOLD
                .space  40*16       ;  fill 0000 between vectors and main program
main:           splk  #0001h,imr     ; enable INT1 interrupt
                clrc   intm
wait:           b wait

 *********Interrupt service routine ISR for HOLD logic*****************************

  int1_hold:
                push                ; Perform any desired context save
                ldp     #0
                in      ICSHDW, IC ; save the contents of IC register
                lacl    #010h       ; load ACC with mask for MODE bit
                and     ICSHDW      ; Filter out all bits except MODE bit
                bcnd    int1,neq    ; Branch if MODE bit is 1, else in HOLD mode
                lacc    imr, 0      ; load ACC with interrupt mask register
                splk    #1, imr     ; mask all interrupts except interrupt1/HOLD
                idle                ; enter HOLD mode, issues HOLDA
                                    ; and the busses will be in tristate
                                    ; wait until rising edge is seen on INT1 pin
                splk    #1, ifr     ; Clear interrupt 1 flag to prevent re entering
                                    ; HOLD mode
                sacl    imr         ; restore interrupt mask register
                pop                 ; Perform necessary context restore

                clrc    intm        ; enable all interrupts
                ret                 ; return from HOLD interrupt

   int1:        nop                 ;  Replace these nops with desired int1
interrupt
                nop                 ; service routine
                pop                 ; Perform necessary context restore****
                clrc    intm        ; enable all interrupts
                ret                 ; return from interrupts
```

4. This IDLE/HOLDA state can be extended until the INT1 pin sees a rising edge. A rising edge on INT1 will force the CPU to exit the IDLE state, deassert the HOLDA signal and return the busses to normal state. Note that the interrupt program code disables all maskable interrupts except the INT1 interrupt to implement safe recovery of HOLDA and busses. Any other sequence of CPU code will cause undesirable bus control and is not recommended.

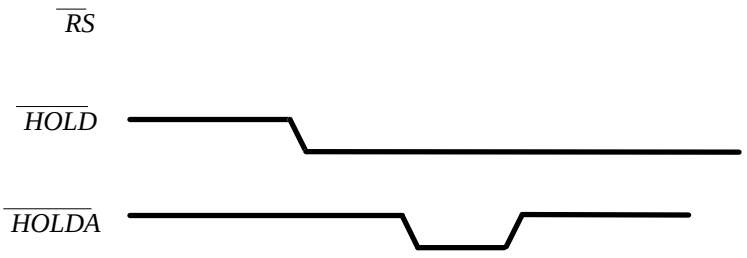
5. Three valid of methods of deasserting the HOLDA and restoring the busses to normal operation are listed below.
   a. Asserting system RESET
   b. Asserting NMI interrupt
   c. Rising interrupt on INT1 in MODE 0.

The first two methods are due to non maskable interrupt sources. If these interrupts occur while in HOLDA, the CPU will deassert HOLDA regardless of the HOLD signal on INT1 pin. The system hardware logic should restore HOLD signal to high state, to avoid further conflicts in the bus control.

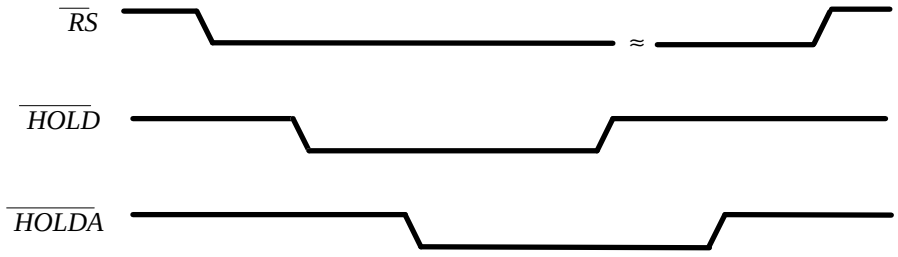
A.2 HOLD on RESET

The HOLD logic can be used to tristate the busses at power-on or reset. This feature is useful in extending the DSP memory control to external processors. If the RESET and HOLD lines are driven low at RESET, the HOLDA will be issued and the busses will be in tristate. The CPU will exit this state on the following conditions.

a. ***Reset deasserted prior to HOLD***
   The CPU will deassert HOLDA regardless of the HOLD signal after the 16 clock cycles required for normal reset operation. Along with HOLDA signal the busses will assume normal state. The external system hardware logic should restore the HOLD signal to high state to avoid conflicts in HOLD logic. Refer to the figure below for details.



b. ***HOLD is asserted prior to RESET***
This is the normal recovery condition HOLD request. Following the HOLD request going high, the HOLDA will be deasserted and the busses will assume normal state.



B. Software Wait State Generator

Due to the fast cycle time of the TMS320C203, it is sometimes necessary to operate with wait states to interface with slower external logic and memory. The software wait state generator can be programmed to generate between zero and seven wait states for a given space. The WSGR has 12 bits: three DATA, six PROGRAM, and three I/O. The wait state generator will insert a wait state(s) to a given memory space based on the value of the three bits, regardless of the condition of the READY signal. The READY signal may be used to extend wait states further. All bits are set to 1 at reset so that the device can operate from slow memory upon reset. The WSGR register resides at I/O port 0xFFFCh.

Wait State Generator Control Register (WSGR).

15 - 12	11 10 9	8 7 6	5 4 3	2 1 0
Reserved	ISWS	DSWS	PSUWS	PSLWS
0	R/W	R/W	R/W	R/`W

Wait state generator values and external READ and WRITE cycles in TMP320C203

Internal Wait state Generator values - 0xfffch	External Read cycles	External Write cycles

0	1	2
1	2	**3 ***
2	3	**3 ***
3	4	4
4	5	5
5	6	6
6	7	7
7	8	8

Note: * - For Wait state generator (WSGR) values 1 and 2 the number of cycles generated for external writes are the same

C. Timer

The timer control register fields are explained below. The TDDR register can now be written to and read.

Timer Control Register (TCR).

15-12	11	10	9-6	5	4	3-0
Reserved	FREE	SOFT	PSC	TRB	TSS	TDDR

Timer Control Register (TCR) Description.

D. Synchronous Serial Port (SSP)

Serial Port Control Register

7	6	5	4	3	2	1	0
OVF	IN0	XRST	RRST	TXM	MCM	FSM	DLB
R	R	R/W	R/W	R/W	R/W	R/W	R/W

Serial Port Control Register Bits Summary

Bit	Name	Function
0	DLB	The Digital Loopback Mode Bit can be used to put the serial port in digital loopback mode. When DLB=1, DR and FSR are connected to DX and FSX, respectively, through multiplexers. Additionally, if CLKR is driven by CLKX if MCM=1. If DLB=1 and MCM=0, CLKR is taken from the CLKR pin of the device. This configuration allows CLKX and CLKR to be tied together externally and supplied by a common external clock source. If DLB=0, DR, FSR, and CLKR are taken from the respective device pins. Note, that TXM must be set to one for proper operation in DLB mode. Note also that the FSX and DX signals appear on the device pins when DLB=1, but FSR and DR do not.
1	FSM	The Frame Synch Mode Bit specifies whether frame synchronization pulses are required for serial port operation. If FSM=1, a frame sync pulse is required on FSX/FSR for the transmission/reception of each word. When the serial port is operated in the continuous mode, FSM=0.
2	MCM	The Clock Mode Bit specifies the clock source for CLKX. If MCM=0, CLKX is taken from the pin. If MCM=1, CLKX is driven by an on–chip clock source having a frequency equal to one–half of CLKOUT1. Note, that if MCM = 1 and DLB = 1, a CLKR signal is also supplied by the internal source.
3	TXM	The Transmit Mode Bit configures the FSX pin as an input (TXM = 0) or as an output (TXM = 1). When TXM = 1, frame sync pulses are generated internally when data is transferred from the DT/R to DSR to initiate data transfers. The internally generated framing signal is synchronous with respect to CLKX. When TXM = 0, the transmitter idles until a frame synch pulse is supplied on the FSX pin.

4 5	RRST XRST	The Transmit Reset and Receive Reset signals reset the transmitter and receiver, respectively. If the SSPCR is to be modified to reconfigure the serial port, a total of two writes should be made to the SSPCR. The first write should write zeroes to XRST and RRST and the desired configuration to bits 0-4. The second write should write ones to XRST and RRST, taking the serial port out of reset. When a zero is written to either of these bits, activity in the corresponding section of the serial port halts. Note that when XRDY=0, writing a zero to XRST1 generates a transmit interrupt. When XRST=0, RRST=0, and MCM=0, the internal clocks to the serial ports are shut off, allowing the device to run in a lower power mode of operation. **The FIFOs will be flushed with zeroes after the first write to SSPCR**.
6	IN0	The Input 0 bit allows the CLKR to be used as a bit input. IN0 reflects the current level of the CLKR pin of the device. The level on this pin can be read by reading the SSPCR register. This bit can be tested by using the BIT or BITT instructions. Note that there is a latency of between 0.5 and 1.5 CLKOUT1 cycles in length from CLKR switching to the new CLKR value being represented in the SSPCR.
7	OVF	Overflow flag (OVF) status signal that indicates when the receiver has finished due to FIFO overflow. The OVF bit is set when the receive FIFO overflows and is cleared when the receive FIFO is read.
8 9	FR0 FR1	FR1 FR0 0 0 Receive FIFO is not empty. 0 1 Receive FIFO has 2 or more words 1 0 Receive FIFO has 3 or 4 words 1 1 Receive FIFO is full.
10 11	FT0 FT1	FT1 FT0 0 0 Transmit FIFO has 1 or more words available. 0 1 Transmit FIFO has 2 or more words available. 1 0 Transmit FIFO has 3 or 4 words available. 1 1 Transmit FIFO is empty. 4 data spaces available.
12	RFNE	Receive FIFO not empty. Data still exists in receive FIFO.
13	TCOMP	Transmit complete. All data has successfully be transferred out of the transmit FIFO.
14	SOFT	The SOFT bit. This bit is enabled when the FREE bit is 0. If FREE = 0, the SOFT bit selects immediate stop if 0, stop after word completion if 1. At reset, this bit is zero.
15	FREE	The FREE bit. If FREE = 1, free run is selected, regardless of the value of the SOFT bit. If FREE = 0, the SOFT bit selects the emulation mode. At reset, this bit is zero.

E. Asynchronous Serial Port (ASP)

Asynchronous Serial Port Control Register (ASPCR)

15	14	13	12 11 10	9	8
FREE	SOFT	URST	Reserved	DIM	TIM
R/W	R/W	R/W	0	R/W	R/W

7	6	5	4	3	2	1	0
RIM	STB	CAD	SETBRK	CIO3	CIO2	CIO1	CIO0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

I/O Status Register (IOSR)

15	14	13	12	11	10	9	8
Reserved	*ADC	*BI	TEMT	*THRE	*FE	*OE	*DR
0	W1C/R	W1C/R	R	R	W1C/R	W1C/R	R

7	6	5	4	3	2	1	0
*DIO3	*DIO2	*DIO1	*DIO0	IO3	IO2	IO1	IO0
W1C/R	W1C/R	W1C/R	W1C/R	**R	**R	**R	**R

Note: W1C translates to write one to clear.
* This bit is ORed to form the interrupt signal from the UART.
** These bits can be written when CIO[3:0] are configured as outputs.

Baud Rate Divisor Register (BRD)

15 0
BRD
R/W

Asynchronous Serial Port Control Register Bit Definitions (ASPCR)

Bit #	Function	Description
0	CIO0	0 = IO0 configured as an input. Default value at reset 1 = IO0 configured as an output.
1	CIO1	0 = IO1 configured as an input. Default value at reset 1 = IO1 configured as an output.

2	CIO2	0 = IO2 configured as an input. Default value at reset 1 = IO2 configured as an output.
3	CIO3	0 = IO3 configured as an input. Default value at reset 1 = IO3 configured as an output.
4	SETBRK	Set break. When set, will force TX output low. When set low, will force TX output high.
5	CAD	Calibrate A detect. Used in auto-baud-alignment.
6	STB	Stop bits. 0 = one stop bit. Default value at reset. 1 = two stop bits.
7	RIM	Receive interrupt mask. A "1" allows receive interrupts (due to BI [break interrupt], FE [framing error], OE [overflow error], or DR [data ready]). A "0" prevents receive interrupts from being generated.
8	TIM	Transmit interrupt mask. A "1" allows transmit interrupts (due to THRE) to be asserted on the UART interrupt signal. A "0" prevents transmit interrupts (THRE) from being asserted.
9	DIM	Delta interrupt mask. If any of the I/O pins change state when DIM is a “1”, a UART interrupt will occur. When DIM is a “0”, this bit has no effect on UART interrupts.
10 11 12	Reserved	Read 0’s.
13	URST	Reset asynchronous serial port. 0 = serial port in reset 1 = write one to enable serial port
14	SOFT	The SOFT bit. This bit is enabled when the FREE bit is 0. If FREE = 0, the SOFT bit selects immediate stop if 0, stop after word completion if 1.
15	FREE	The FREE bit. If FREE = 1, free run is selected, regardless of the value of the SOFT bit. If FREE = 0, the SOFT bit selects the emulation mode.

I/O STATUS REGISTER (IOSR)

Bit #	Function	Description
0	IO0	Current level on IO0.
1	IO1	Current level on IO1.
2	IO2	Current level on IO2.
3	IO3	Current level on IO3.
4	DIO0	change detected on pin IO0. Change only detected when IO0 is configured as an input. Write a one to this bit to clear it.
5	DIO1	change detected on pin IO1. Change only detected when IO1 is configured as an input. Write a one to this bit to clear it.
6	DIO2	change detected on pin IO2. Change only detected when IO2 is configured as an input. Write a one to this bit to clear it.
7	DIO3	change detected on pin IO3. Change only detected when IO3 is configured as an input. Write a one to this bit to clear it.
8	DR	Data ready indicator for the receiver. 0 = Normal operation. 1 = Set when a complete incoming character has been received and transferred into the receiver register (ADTR). Reset to logic 0 when receive register (ADTR) is read. Cleared to zero on reset.
9	OE	Receiver register (ADTR) overrun indicator. 0 = Normal operation. 1 = Before the character in the receive register (ADTR) was read, it was overwritten by the next character transferred into the register. The OE bit is cleared by writing 1 to the bit. Cleared to 0 on reset.
10	FE	Framing error indicator. 0 = Normal operation. No framing error detected. 1 = Indicates that the received character did not have a valid (logic 1) stop bit. The FE bit is cleared by writing 1 to the bit. Cleared to 0 on reset.
11	THRE	Transmitter register (ADTR) empty indicator. 0 = Normal operation. 1 = When the transmitter register (ADTR) is empty, indicating that the UART is ready to accept a new character. THRE is set to one when the contents of the transmitter register (ADTR) are transferred to the transmitter shift register (AXSR). This bit is reset to 0 concurrent with the loading of the transmitter register (ADTR) by the CPU.

12	TEMT	Transmit empty indicator. 0 = Transmitter register (ADTR) and/or transmitter shift register (AXSR) are full. 1 = Transmit register (ADTR) and transmit shift register (AXSR) are both empty. When either the transmitter register (ADTR) or the transmitter shift register (AXSR) contains a data character, the TEMT bit is reset to logic 0. Cleared to 0 on reset.
13	BI	Break interrupt. Indicates a break has been detected on RX. Write 1 to clear. Cleared on Reset
14	ADC	A detect complete. Used in auto-baud-alignment. This bit gets set, while CAD bit is 1 and Character 'A or a ' is received in ADTR. The ADTR should be read after ADC is set to avoid any subsequent OE. Write 1 to clear ADC bit. An auto baud detect will cause an interrupt, while DIM TIM, RIM bits are disabled or set to 0.
15	Reserved	Read 0.

E.1 Auto baud detection

The auto baud detection logic is functional on the asynchronous serial port. The following steps explain the sequence by which the detection logic could be implemented.

1. Enable auto baud detection by setting the CAD bit in ASPCR to 1 and ADC bit in IOSR to zero.

2. Receive character " A" or " a" as the first character at any desired baud rate definable in the BRD register. If the first character received is any one of the above, the serial port will lock to the incoming baud rate and the BRD register will be updated to the incoming baud rate value.

3. Baud rate detection is indicated by an TXRXINT (0x000c) interrupt, if the IMR bit is set to enable the asynchronous port interrupt (0x0020h). This interrupt will occur regardless of the DIM/TIM/RIM bits in the ASPCR.

4. Following the baud detection interrupt, the ADTR should be read to clear the receive buffer. If the ADTR is not cleared, any subsequent character received
will set the OE bit in IOSR, indicating an overrun error.

5. Once the baud rate is detected, both the CAD and ADC bits must be cleared. Write 0 to CDC to clear it and write 1 to clear ADC. If these bits are not cleared the auto baud detection will try to lock to the incoming character speed.