HW06-1(Exceptions-General)

Exceptions:

- come in one of two types: **interrupts** and **resets**
- can be caused by illegal or erroneous instructions **software**
- is a break in normal program flow.
- can be divided into two types: **Hardware** interrupts, e.g an external interrupt when a pin changes, and **Software** interrupts, e.g. a divide by zero occurs"

Interrupts can be divided into two types: **asynchronous** which can happen at any time, and **synchronous** which occur at predetermined intervals.

Trying to execute an illegal instruction produces a **software** interrupt.

Another name for a software interrupt is a **trap** because it can catch a problem with a software instruction

Why are interrupts so important when dealing with I/O? That is, what do interrupts allow that polling does not
interrupt will stop. polling will continue even though an interrupt was detected.

A **interrupt** is an exception which saves the current processor state before executing the service routine

Which of the following are potential sources for maskable interrupts? **port H,J,P,PWM,SPI,SCI**

There are two types of interrupts which are determined by their timing.
An example if an **Asynchronous** interrupt is when a process detects an error condition.
An example of a **Synchronous** interrupt is a timer used to write to a display at specified periodic intervals.

There are two types of interrupts which are determined by their timing. **Synchronous** interrupts can be used to switch between processes at predetermined intervals.

A **Maskable** interrupt can be disabled and enabled through software.

Interrupts can be divided into two types: **Maskable** which can be turned on or off through the clearing or setting of certain register bits, and **Non-maskable** which can be turned on in software, but cannot be turned off with software.

An **Interrupt** is an exception which saves the current processor state before executing the service routine.

What is a clock monitor reset?
When a clock monitor senses a clock error and resets the clock?

What is the difference between a power-on reset and an external reset?
While they are both triggered by low-active reset pins, the Power-On Reset is activated at startup to guarantee a known initial state while the external reset is triggered by an external switch.

What is a Power-On Reset (POR)?
A low-active reset pin; it is activated at startup and guarantees a known initial state.

Hardware exceptions can be caused by resets initiated by the user or processor or caused by external interrupts and internal timers.

What does the programmer need to do every time an interrupt is handled by an interrupt service routine?

A[n] **interrupt** is an exception which saves the current processor state before executing the service routine.

A[n] **reset** is an exception which does not save the state of processor before executing its service routine.

A **clock monitor** reset occurs when the processor detects the system clock is not within a valid frequency range.

A[n] example of an **External** Reset is when a low-active pin is activated by a user pressing a switch to force the processor to reset.

A[n] **reset** is an exception described as a "one-way ticket" because the processor doesn't return to what it was doing when the exception occurred.

There are two types of interrupts which are determined by their timing. An example an **asynchronous** interrupt is when a processor detects an error condition.

What is a Watchdog Reset? A reset that happens if a signal has not been sent to the watchdog to reset the timer in a given amount of time.

What is a COP Reset? Same as above

Hardware exceptions can be caused by resets initiated by the user or processor or caused by external interrupts and internal timers.

Software exceptions can be caused by illegal or erroneous instructions

HW06-2(PIC32MX-Resets)

- Give a Single C instruction which will set variable **n** to a **1** if a PIC32MX was *not* in Sleep Mode when it was reset.
  if (RCONbits.SLEEP==0)
  n = 1;
  if (!RCONbits.SLEEP)
  n = 1;
- Give a Single C instruction which will set variable **n** to a **1** if a PIC32MX was in Sleep Mode when it was reset.
  All of these are valid answers:
  if (RCONbits.SLEEP==1) n = 1;
  if (RCONbits.SLEEP==1) n=1;
  if (RCONbits.SLEEP==1) n = 1
  if (RCONbits.SLEEP==1) n=1
  if (RCON & 0x0008) n = 1;
  if (RCONbits.SLEEP) n = 1;
  if (RCONbits.SLEEP) n=1;
  if (RCONbits.SLEEP) n = 1
  if (RCONbits.SLEEP) n=1
- Give a single C instruction which will set variable **n** to a **1** if a software reset has occurred on a PIC32MX.
  if (RCONbits.SWR==1) n = 1;
  if (RCONbits.SWR) n = 1;
  if (RCON & 0x0040) n = 1;
- Give a single C instruction which will enable a Software Reset on a PIC32MX.
  RSWRSTbits.SWRST = 1;
- Give a single C instructions which will disable a Software Reset on a PIC32MX.
  RSWRSTbits.SWRST = 0;
- Give a single C instructions which will set variable **n** to a **1** if a Power-On Reset has *not* occurred on a PIC32MX.
  if (RCONbits.POR==0) n = 1;
  if (RCONbits.POR==0) n=1;
  if (RCONbits.POR==0) n = 1
  if (RCONbits.POR==0) n=1
  if (!(RCON & 0x0001)) n = 1;
  if (!RCONbits.POR) n = 1;
  if (!RCONbits.POR) n=1;
  if (!RCONbits.POR) n = 1
  if (!RCONbits.POR) n=1
- Give a single C instructions which will set variable n to a 1 if a Brown-Out Reset has *not* occurred on a PIC32MX.
  if (RCONbits.BOR==0) n = 1;
- Give a single C instructions which will set variable n to a 1 if a Watchdog Timeout Reset has not occurred on a PIC32MX.
  if (RCONbits.WDTO==0) n = 1;
  if (RCONbits.WDTO==0) n=1;
  if (RCONbits.WDTO==0) n = 1
  if (RCONbits.WDTO==0) n=1
  if (!(RCON & 0x0010)) n = 1;
  if (!RCONbits.WDTO) n = 1;
  if (!RCONbits.WDTO) n=1;
  if (!RCONbits.WDTO) n = 1
  if (!RCONbits.WDTO) n=1
- Give a single C instructions which will set variable n to a 1 if a PIC32MX was in Idle Mode when it was reset.
  if (RCONbits.IDLE==1) n = 1;
  if (RCONbits.IDLE==1) n=1;

HW06-3(PIC32-InterruptInit)

External Interrupt	.INTx
Input Capture	.ICx
Comparator	CMPx

Write the C instructions necessary to initialize the External Interrupt 2 interrupt on a PIC32MX by:

1. Clearing the flag
2. Setting the priority to level 5
3. Setting the sub-priority to level 0
4. Enabling the interrupt.

1. IFS0bits.INT2IF = 0;
2. IPC2bits.INT2IP = 5;
3. IPC2bits.INT2IS = 0;
4. IEC0bits.INT2IE = 1;

Write the C instructions necessary to initialize the Output Compare 3 interrupt on a PIC32MX by:

1. Clearing the interrupt flag.
2. Setting the priority to level 5.
3. Setting the sub-priority to level 0.
4. Enabling the interrupt.
   1. **IFS0bits.OC3IF = 0;**
   2. **IPC3bits.OC3IP = 5;**
   3. **IPC3bits.OC3IS = 0;**
   4. **IEC0bits.OC3IE = 1;**

Write the C instructions necessary to initialize the I2C2 Slave Event interrupt on a PIC32MX by:

1. Clearing the interrupt flag.
2. Setting the priority to level 6.
3. Setting the sub-priority to level 1.
4. Enabling the interrupt.

   1. **IFS1bits.I2C2SIF = 0;**
   2. **IPC9bits.I2C2IP = 6;**
   3. **IPC9bits.I2C2IS = 1;**
   4. **IEC1bits.I2C2SIE = 1;**

Write the C instructions necessary to initIFialize the I2C1 Bus Collision Event interrupt on a PIC32MX by:

1. Clearing the interrupt flag. **IFS1bits.I2C1BIF = 0;**
2. Setting the priority to level 4. **IPC8bits.I2C1IP = 4;**
3. Setting the sub-priority to level 0. **IPC8bits.I2C1IS = 0;**
4. Enabling the interrupt. **IEC1bits.I2C1BIE = 1;**

Write the C instructions necessary to initialize the SPI1 Transfer Done interrupt on a PIC32MX by:

1. Clearing the interrrupt flag. **IFS1bits.SPI1TXIF = 0;**
2. Setting the priority to level 6. **IPC7bits.SPI1IP = 6;**
3. Setting the sub-priority to level 1. **IPC7bits.SPI1IS = 1;**
4. Enabling the interrupt. **IEC1bits.SPI1TXIE = 1;**

Write the C instructions necessary to initialize the I2C1 Master Event interrupt on a PIC32MX by:

1. Clearing the interrrupt flag. **IFS1bits.I2C1MIF = 0;**
2. Setting the priority to level 3. **IPC8bits.I2C1IP = 3**;
3. Setting the sub-priority to level 0. **IPC8bits.I2C1IS = 0;**
4. Enabling the interrupt. **IEC1bits.I2C1MIE = 1;**

Write the C instructions necessary to initialize the SPI2 Fault interrupt on a PIC32MX by:

1. Clearing the interrrupt flag. **IFS1bits.SPI2EIF = 0;**
2. Setting the priority to level 1. **IPC9bits.SPI2IP = 1;**
3. Setting the sub-priority to level 3. **IPC9bits.SPI2IS = 3;**

Write the C instructions necessary to initialize the Input Capture 1 interrupt on a PIC32MX by:

1. Clearing the flag
2. Setting the priority to level 1
3. Setting the sub-priority to level 3
4. Enabling the interrupt.

1. IFS0bits.IC1IF = 0;
2. IPC1bits.IC1IP = 1;
3. IPC1bits.IC1IS = 3;
4. IEC0bits.IC1E = 1;

Write the C instructions necessary to initialize the Comparator Interrupt 2 on a PIC32MX by:

1. Clearing the flag
2. Setting the priority to level 1
3. Setting the sub-priority to level 3
4. Enabling the interrupt.

1. IFS1bits.CMP2IF = 0;
2. IPC7bits.CMP2IP = 0;
3. IPC7bits.CMP2IS = 2;
4. IEC1bits.CMP2IE = 1;

Write the C instructions necessary to initialize the External Interrupt 4 interrupt on a PIC32MX by:

1. Clearing the interrupt flag.
2. Setting the priority to level 5.
3. Setting the sub-priority to level 3.
4. Enabling the interrupt.
   1. **IFS0bits.INT4IF = 0;**
   2. **IPC4bits.INT4IP = 5;**
   3. **IPC4bits.INT4IS = 3;**
   4. **IEC0bits.INT4IE = 1;**

Write the C instructions necessary to initialize the Timer3 interrupt on a PIC32MX by:

1. Clearing the interrupt flag.
2. Setting the priority to level 6.
3. Setting the sub-priority to level 3.
4. Enabling the interrupt.
   1.IFS0bits.T3IF = 0;
   2.IPC3bits.T3IP = 6;
   3.IPC3bits.T3IS = 3;
   4.IEC0bits.T3IE = 1;

Write the C instructions necessary to initialize the Timer2 interrupt on a PIC32MX by:

1. Clearing the interrupt flag.
2. Setting the priority to level 3.
3. Setting the sub-priority to level 0.
4. Enabling the interrupt.
   1.IFS0bits.T2IF = 0;
   2.IPC2bits.T2IP = 3;
   3.IPC2bits.T2IS = 0;
   4.IEC0bits.T2IE = 1;

HW06-3(PIC32-ExternalInterruptInit)
Write the C instruction necessary to have the external interrupt INT1 on a PIC32MX look for a Falling edge.
INTCONbits.INT1EP = 0;

..For INT4:
INTCONbits.INT4EP = 0;

HW06-3(PIC32-InterruptInit2)
See Table 7-1 for reference.
Write the C instructions to initialize the Input Capture 4 Error interrupt on a PIC32MX by:

1. Clearing the flag
2. Setting the priority to level 1
3. Setting the sub-priority to level 3
4. Enabling the interrupt.

1. IFS0bits.IC4EIF = 0;
2. IPC4bits.IC4IP = 7;
3. IPC4bits.IC4IS = 3;
4. IEC0bits.IC4IE = 1;

Write the C instructions necessary to initialize the PORTC Input Change Interrupt interrupt on a PIC32MX by:

1. Clearing the interrupt flag.
2. Setting the priority to level 1.
3. Setting the sub-priority to level 2.
4. Enabling the interrupt.
   1. **IFS1bits.CNCIF = 0;**
   2. **IPC8bits.CNIP = 1;**
   3. **IPC8bits.CNIS = 2;**
   4. **IEC1bits.CNCIE = 1;**

Write the C instructions necessary to initialize the PORTA Input Change Interrupt interrupt on a PIC32MX by:

1. Clearing the interrupt flag. **IFS1bits.CNAIF = 0;**
2. Setting the priority to level 0. **IPC8bits.CNIP = 0;**
3. Setting the sub-priority to level 3. **IPC8bits.CNIS = 3;**
4. Enabling the interrupt. **IEC1bits.CNAIE = 1;**

Write the C instructions necessary to initialize the UART1 Fault interrupt on a PIC32MX by:**WHY IS THIS "C8" AND NOT "C6"?**

1. Clearing the flag. **IFS1bits.U2RXIF = 0;**
2. Setting the priority to level 5. **IPC9bits.U2IP = 1;**
3. Setting the sub-priority to level 0. **IPC9bits.U2IS = 2;**
4. Enabling the interrupt. **IEC1bits.U2RXIE = 1;**

Write the C instructions necessary to initialize the UART2 Receiver interrupt on a PIC32MX by:

1. Clearing the flag. **IFS1bits.U1EIF = 0;**
2. Setting the priority to level 5. **IPC8bits.U1IP = 1;**
3. Setting the sub-priority to level 0. **IPC8bits.U1IS = 0;**
4. Enabling the interrupt. **IEC1bits.U1EIE = 1;**

Write the C instructions necessary to initialize the UART2 Transmitter interrupt on a PIC32MX by:

1. Clearing the interrupt flag. **IFS1bits.U2TXIF = 0;**
2. Setting the priority to level 5. **IPC9bits.U2IP = 5;**
3. Setting the sub-priority to level 1. **IPC9bits.U2IS = 1;**
4. Enabling the interrupt. **IEC1bits.U2TXIE = 1;**

Write the C instructions necessary to initialize the I2C1 Slave Event interrupt on a PIC32MX by:

1. Clearing the flag
2. Setting the priority to level 6
3. Setting the sub-priority to level 0
4. Enabling the interrupt
   1. IFS1bits.I2C1SIF = 0;
   2. IPC8bits.I2C1IP = 6;
   3. IPC8bits.I2C1IS = 0;
   4. IEC1bits.I2C1SIE = 1;

4. Enabling the interrupt. **IEC1bits.SPI2EIE = 1;**

Write the C instructions necessary to initialize the Core Software Interrupt 1 on a PIC32MX by:

1. Clearing the interrupt flag. **IFS0bits.CS1IF = 0;**
2. Setting the priority to level 1. **IPC0bits.CS1IP = 1;**
3. Setting the sub-priority to level 3. **IPC0bits.CS1IS = 0;**
4. Enabling the interrupt. **IEC0bits.CS1IE = 1;**

Write the C instructions necessary to initialize the Flash Control Event on a PIC32MX by:

1. Clearing the interrupt flag. **IFS0bits.FCEIF = 0;**
2. Setting the priority to level 1. **IPC6bits.FCEIP = 1;**
3. Setting the sub-priority to level 3. **IPC6bits.FCEIS = 0;**
4. Enabling the interrupt. **IEC0bits.FCEIE = 1;**

Write the C instructions necessary to initialize the External Interrupt 1 interrupt on a PIC32MX by:

1. Clearing the interrupt flag. **IFS0bits.INT1IF = 0;**
2. Setting the priority to level 3. **IPC1bits.INT1IP = 3;**
3. Setting the sub-priority to level 2. **IPC1bits.INT1IS = 2;**
4. Enabling the interrupt. **IEC0bits.INT1IE = 1;**

Write the C instructions necessary to initialize the Output Compare 1 interrupt on a PIC32MX by:

1. Clearing the interrupt flag. **IFS0bits.OC1IF = 0;**
2. Setting the priority to level 1. **IPC1bits.OC1IP = 1;**
3. Setting the sub-priority to level 3. **IPC1bits.OC1IS = 3;**
4. Enabling the interrupt **IEC0bits.OC1IE = 1;**

HW06-3(PIC32MX-Interrupts)
How many different Priority values are there in the sub-priority level of a PIC32MX? **4**
What is the *lowest* priority *value* in the *sub*-priority level of a PIC32MX? **0**

How is a persistent interrupt different form a non-persistent interrupt?
Persistent interrupts will remain active and the associated interrupt flag set until the issue causing the interrupt serviced. In non persistent interrupts, the interrupt is recorded once to the interrupt controller which presents it to the CPU.

What is the highest priority value in the main priority level? **7**

How many different priority *levels* are there in the PIC32MX? **2**

What header file provides functions and definitions to handle PIC32MX interrupts? **int.h**

what is the lowest priority value in the main priority level? **0**

what external events can cause an external interrupt? **a reset**

How many possible interrupt sources does a PIC32 have? **64**

What does the INTSTAT register in the PIC32 tell the programmer?
The priority number and vector number of the latest interrupt presented to the CPU. - Pg 91 Data sheet
what does the IP17 mean in the code below?
#pragma interrupt Int0_IRQ ip17 vector 3

What does the **ipl2** mean in the code below?

```
void __ISR(_EXTERNAL_0_VECTOR,
ipl2) Int0_IRQ(void);
```

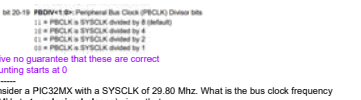
What external event(s) can cause an external interrupt on a PIC32MX?
The low active reset pin being activated (set HIGH) *This isn't a confirmed answer*

What does a PIC32MX do if two interrupts with the *same* priority value occur at the same time?
it compares the sub priorities
Chapter 7

HW07-2(PIC32-OSCCON-Frequency)
Consider a PIC32MX with a SYSCLK of 34.20 Mhz. What is the bus clock frequency (in MHz to two decimal places) given that
OSCCON = 0x1950724F
PBCLK = 8.55 MHz

Instructions:

1. Take the bits 19 and 20 from the binary value of OSCCON:
   1100101010**00**00111001001001111 (start count from right to left, starting from 0)
2. Refer to the chart/image below (found on pg. 97 of datasheet) & take the frequency given and divide by whatever is says in the image to get answer. In this case: 34.20 MHz ÷ 4 = 8.55 MHz



*I give no guarantee that these are correct
*counting starts at 0

Consider a PIC32MX with a SYSCLK of 29.80 Mhz. What is the bus clock frequency (in MHz to **two decimal places**) given that
OSCCON = 0x394172DD
PBCLK = 29.80 MHz

Consider a PIC32MX with a SYSCLK of 15.60 Mhz. What is the bus clock frequency (in MHz to two decimal places) given that
OSCCON = 0x2A7E12A3
PBCLK = 1.95 MHz

Consider a PIC32MX with a SYSCLK of 10.10 Mhz. What is the bus clock frequency (in MHz to two decimal places) given that
OSCCON = 0x027842B4
PBCLK = 1.26 MHz

Consider a PIC32MX with a SYSCLK of 14.30 Mhz. What is the bus clock frequency (in MHz to two decimal places) given that
OSCCON = 0x07023278
PBCLK = 14.3 MHz

HW07-2(PIC32-OSCCON-PBDIV)
Consider a PIC32MX with a SYSCLK of 20.90 MHz. Give a single C instruction to produce a bus clock as close to 14.72 MHz as possible.
OSCCONbits.PBDIV = 1;

I can't find the exact steps on how to do this, so here is my best guess:

1. Take the SYSCLK frequency and divide it by 1,2,4, and 8. You can do this all at once in your calculator by typing: `20.90/{1,2,4,8}`
   a. This should give you 4 results in brackets: `{20.9 10.45 5.225 2.6125}`
   b. All this command does is divide one number by multiple numbers all at once.
2. Take the result that is closest to the desired bus clock frequency, in this case 10.45 is closest to 14.72MHz, and note which number you divided it by. In this case, 10.45 was the result of 20.90/2, so that divisor was 2.
3. Using the chart used in the previous question (7-2 OSCCON-Frequency), take the divisor bits that match up with your value determined in step 2 and convert to decimal. In this example, the line that says "...divided by 2", has divisor bits 01, which is just decimal value 1. This means you need to set this command equal to

---

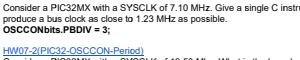
Consider a PIC32MX with a SYSCLK of 26.60 MHz. Give a single C instruction to produce a bus clock as close to 5.20 MHz as possible.
OSCCONbits.PBDIV = 2;
`26.60/{1,2,4,8} = {26.6 13.3 6.65 3.325}`

Consider a PIC32MX with a SYSCLK of 11.40 MHz. Give a single C instruction to produce a bus clock as close to 3.96 MHz as possible.
OSCCONbits.PBDIV = 2;

Consider a PIC32MX with a SYSCLK of 11.70 MHz. Give a single C instruction to produce a bus clock as close to 4.53 MHz as possible.
OSCCONbits.PBDIV = 1;

Consider a PIC32MX with a SYSCLK of 7.10 MHz. Give a single C instruction to produce a bus clock as close to 1.23 MHz as possible.
OSCCONbits.PBDIV = 3;

HW07-2(PIC32-OSCCON-Period)
Consider a PIC32MX with a SYSCLK of 19.50 Mhz. What is the bus clock period (in ns to 2 decimal places) given that OSCCON = 0x2B04326B
Period of PBCLK = **51.28** ns



Use the same chart from the last two 7-2 problems.
EQUATION:
(n=number in "divided by n", F=SYSCLK frequency (**in Hz**)
Period = 10^9/(F*10^6 * (1/n))

1. Convert hex number to binary and get bits 20-19 to figure out what to divide SYSCLK by, this is your value for *n*.
2. Answer = 10^9/(19.5*10^6 * (1/1)) = 51.28205 ns = 51.28 ns

Consider a PIC32MX with a SYSCLK of 37.40 Mhz. What is the bus clock period (in ns to 2 decimal places) given that OSCCON = 0x200F02D1
Period of PBCLK = **53.48** ns

Consider a PIC32MX with a SYSCLK of 24.00 Mhz. What is the bus clock period (in ns to 2 decimal places) given that OSCCON = 0x166E026B
Period of PBCLK = **83.33** ns

Consider a PIC32MX with a SYSCLK of 6.20 Mhz. What is the bus clock period (in ns to 2 decimal places) given that OSCCON = 0x135622F8
Period of PBCLK = **645.16** ns

Consider a PIC32MX with a SYSCLK of 15.60 Mhz. What is the bus clock period (*in ns to 2 decimal places*) given that OSCCON = 0x3A25124A
Period of PBCLK = **64.10** ns

HW07-3(PIC32-CoreTimer-CompareForFrequency)

Given the following C instructions on a PIC32MX running at 10.5 MHz,

```
_CP0_SET_COUNT(0);
_CP0_SET_COMPARE(compare);
IFS0bits.CTIF = 0;
IFS0bits.CTIE = 1;
```

what does the value of compare need to be to give a Core Timer interrupt frequency as close to 0.656 Hz as possible?
Answer = **8003049**

Formula: $f_{CT}=f_{SYSCLK}/(2 * COMPARE)$
Using the above problem, solve for COMPARE: 0.656 = 10.5*10^6 / (2 * COMPARE), COMPARE = 8003049.
I use the solve() function on the TI-89 to avoid confusion, and less work. On the TI-89 you can just enter: `solve(0.656=10.5e6/(2*x),x)`

Given the following C instructions on a PIC32MX running at 6.8 MHz,

```
_CP0_SET_COUNT(0);
_CP0_SET_COMPARE(compare);
IFS0bits.CTIF = 0;
IFS0bits.CTIE = 1;
```

what does the value of compare need to be to give a Core Timer interrupt frequency as close to 701 Hz as possible?
Answer = **4850**

Given the following C instructions on a PIC32MX running at 19.4 MHz,

```
_CP0_SET_COUNT(0);
_CP0_SET_COMPARE(compare);
IFS0bits.CTIF = 0;
IFS0bits.CTIE = 1;
```

what does the value of compare need to be to give a Core Timer interrupt frequency as close to 2.4 MHz as possible?
Answer = **4.0417**

HW07-3(PIC32-CoreTimer-CompareForPeriod)
Formula: $f_{CT}=f_{SYSCLK}/(2 * COMPARE)$
Given the following C instructions on a PIC32MX running at 19.6 MHz,

```
_CP0_SET_COUNT(0);
_CP0_SET_COMPARE(compare);
IFS0bits.CTIF = 0;
IFS0bits.CTIE = 1;
```

what does the value of compare need to be to give a Core Timer interrupt frequency as close to 183 kHz as possible?
Answer = **53.55**
Same process as CompareForFrequency above.
If given time, f=1/T

Given the following C instructions on a PIC32MX running at 2.1 MHz,

```
_CP0_SET_COUNT(0);
_CP0_SET_COMPARE(compare);
IFS0bits.CTIF = 0;
IFS0bits.CTIE = 1;
```

what does the value of compare need to be to give a Core Timer interrupt frequency as close to 363 kHz as possible?
Answer = **2892.56**

HW07-3(PIC32-CoreTimer-WhatIsFrequency)
Given the following C instructions on a PIC32MX running at 13 MHz,

```
_CP0_SET_COUNT(0);
_CP0_SET_COMPARE(104202);
IFS0bits.CTIF = 0;
IFS0bits.CTIE = 1;
```

what is the frequency of the Core Timer interrupt in Hz (to one decimal place)?
Answer = **62.4**
Same equation as CompareForFrequency and CompareForPeriod above, but you're solving for f_{CT} this time. These problems seem to be picky about the decimal places *sometimes*.Formula: $f_{CT}=f_{SYSCLK}/(2 * COMPARE)$

HW07-3(PIC32-CoreTimer-WhatIsPeriod)
Given the following C instructions on a PIC32MX running at 3.9 MHz,

```
_CP0_SET_COUNT(0);
_CP0_SET_COMPARE(220);
IFS0bits.CTIF = 0;
IFS0bits.CTIE = 1;
```

what is the period of the Core Timer interrupt in **micro** seconds (to one decimal place)?
Answer = **112.8**

Instructions
1. Use the same formula as above: $f_{CT}=f_{SYSCLK}/(2 * COMPARE)$
   a. To get T_{CT} (period of core timer) in micro seconds: $T_{CT} = 10^6/f_{CT}$
   b. To get T_{CT} in **milli**seconds: $T_{CT} = 10^3/f_{CT}$

Given the following C instructions on a PIC32MX running at 3 MHz,

```
_CP0_SET_COUNT(0);
_CP0_SET_COMPARE(859590);
IFS0bits.CTIF = 0;
IFS0bits.CTIE = 1;
0
```

what is the period of the Core Timer interrupt in **milliseconds** (to one decimal place)?
Answer = **573.1**

HW07-4(PIC32-T1CON)
Give a single C instruction to configure a PIC32MX's Timer 1 with the following properties.

Timer is enabled
Continue module operation when the device enters Idle Mode.
Writes to Timer1 are ignored until pending write operation completes.
1:64 prescale value

External clock input IS synchronized.
External clock.

Answer: **T1CON = 0x9026;**

Instructions:
See page 157 of the datasheet for Timers 2-5 and page 152 for Timer 1 for reference/further info
1. Build the binary number: 0000000000000000**A**0**CDE**000**G**0**II**0**KL**0
   - A = Timer on bit (1=on)
   - C = SIDL (1=discontinue operation in idle mode)
   - D = TWDIS (1=writes to timer are ignored until pending write is complete)
   - E = TWIP (1=asynchronous write to timer1 is in progress)
   - G = TGATE *This bit depends on the TCS bit (L). If ignored, =0.*
   - II = TCKPS (2 bits) (00=1,01=8,10=64,11=256)
   - K = TSYNC (1=external clock input is synchronized
   - L = TCS (1=external clock)
2. For this problem, this would leave you with this (with leading zeros excluded): 1001000000**1**0**1**0 (Note: I underlined the middle zeros to make this easier)
3. Convert that binary number to hex and make it into the command.

Give a single C instruction to configure a PIC32MX's Timer 1 with the following properties.

Timer is disabled.
Discontinue module operation when the device enters Idle Mode.
Back-to-back writes are enabled.
1:64 prescale value
External clock input IS synchronized.
External clock.

Answer = **T1CON = 0x2026;**
Steps I took to "build" this binary number:
1. 0-100---0-10-11-
2. 0010000000100110
3. In hex: 0x2026

HW07-4(PIC32-TimerClockFrequency)
A PIC32MX has a peripheral bus clock frequency of 21.4 MHz.
What is the Timer 4 clock frequency (in MHz to three decimal places) given the following register setting?

T4CON = 0x00008010
Answer = **10.700;**

Note: Timer 1 is a type A and timers 2-5 are type B timers, but I don't think that is important for this. -It matters for which bits you look at, Type A is only 2 bits, B is 3
1. Convert hex to binary
2. Get bits 4-6 and look at TCKPS value and find the prescale (PS) value
3. Plug that into this equation: $C_{BUS} = C_{CORE}$ / PS and solve for C_{CORE} (Core clock frequency)

That will look like this: `21.4e6 * (1/2) * 10^-6 = 10.700`

T1CON, TYPE A TCKPS 7-4 CHART

bit 5-4 **TCKPS<1:0>:** Timer Input Clock Prescale Select bits
11 = 1:256 prescale value
10 = 1:64 prescale value
01 = 1:8 prescale value
00 = 1:1 prescale value

TxCON, TYPE B TCKPS

bit 6-4 **TCKPS<2:0>:** Timer Input Clock Prescale Select bits(3)
111 = 1:256 prescale value
110 = 1:64 prescale value
101 = 1:32 prescale value
100 = 1:16 prescale value
011 = 1:8 prescale value
010 = 1:4 prescale value
001 = 1:2 prescale value
000 = 1:1 prescale value

A PIC32MX has a peripheral bus clock frequency of 6.3 MHz.
What is the Timer 4 clock frequency (in MHz to three decimal places) given the following register setting?

T4CON = 0x00008010;
Answer = **3.150**

HW07-4(PIC32-TimerClockPeriod)
A PIC32MX has a peripheral bus clock frequency of 30.1 MHz
What is the Timer 1 clock period (in µs to three decimal places) given the following register setting?
T1CON = 0x000088B0;
Answer = **8.505**

Similar to the previous 7-4 problem, get the binary value of the hex number and read the prescale (PS) value from the *7-4 Chart*. Use the equation below to get the result in microseconds. Remember PS is a fraction (e.g. ¼ or 1/256)

$T_{CLK} = 10^6/(C_{BUS} \times PS)$
Using above problem: `10^6/(30.1e6*1/256)`

A PIC32MX has a peripheral bus clock frequency of 6.3 MHz.
What is the Timer 3 clock period (in µs to three decimal places) given the following register setting?
T3CON = 0x00000008;
Answer = **0.159**

HW07-4(PIC32-TimerPeriod-MinMax)
If a PIC32MX has a bus clock of 16.80 MHz, what does TCKPS need to be to be able to produce timer periods from 0.07 µs to 3.8 ms with Timer 4? (Type 'X' if not possible.)
Answer = **0**

Use the equation in TimerClockPeriod, but solve for PS. *WORK IN BASE UNITS*
$T_{CLK} = 10^6/(C_{BUS} \times PS)$
1. A = PS using minimum timer period (0.07 us)
2. B = PS using max timer period (3.8 ms)
3. Choose the prescale value from *Chart 7-4* that is just smaller than the min calculated between A and B.
4. The binary value (bits 6-4) associated with the prescale value is converted to decimal and is the final answer
5. If B is greater than 65536 (2^{16}). The answer is **X**
6. If B is less than 65536, then you match it up with the closest possible prescale value in the *7-4 Chart* and that should be your answer. I haven't gotten anything other than 0 or X though.

You can also just use this equation in the calculator to get A & B all at once:
`csolve(1/(C_BUS*1/n)={T_MIN, T_MAX},n)`. Example: `csolve(1/(16.6e6*1/n)={.07e-6,3.8e-3})` Result: `n={1.162, 63080}`
Check if Possible

```
      TMin = 1/bus            TMax = (1/bus)*(2^16-1)
if   Tmin < min    &         Tmax > max ?       ans is not X
```

If a PIC32MX has a bus clock of 5.10 MHz, what does TCKPS need to be to be able to produce timer periods from 0.20 µs to 10 ms with Timer 1? (Type 'X' if not possible.)
Answer = **{1.02, 51000}**

If a PIC32MX has a bus clock of 15.10 MHz, what does TCKPS need to be to be able to produce timer periods from 0.07 µs to 4.4 ms with Timer 3? (Type 'X' if not possible.)
Answer = **X** {1.057, 66440} Tmax = .0043 is less than .0044 => X

If a PIC32MX has a bus clock of 18.90 MHz, what does TCKPS need to be to be able to produce timer periods from 0.06 µs to 3.4 ms with Timer 1? (Type 'X' if not possible.)
Answer = **0** {1.134, 64260} Tmax = 0.00346 is greater than 0.0034 => not X

If a PIC32MX has a bus clock of 10.20 MHz, what does TCKPS need to be to be able to produce timer periods from 26 µs to 1.7 seconds with Timer 2? (Type 'X' if not possible.)
Answer = **X** {265.2, 17340000}

If a PIC32MX has a bus clock of 18.90 MHz, what does TCKPS need to be to be able to produce timer periods from 15 µs to .85 seconds with Timer 4? (Type 'X' if not possible.)
Answer = **7** {283.5, 16065000} doesn't work?
Can someone explain why this is 7 and not X? Is it because the PS is 256 and 256*65536 is the new value MAX should be compared too? *It's a mystery.*

If a PIC32MX has a bus clock of **19.80 MHz**, what does **TCKPS** need to be to be able to produce timer periods from **13 µs** to **0.80 seconds** with **Timer 4**? (Type 'X' if not possible.)
Answer = **7**

HW07-4(PIC32-TimerPeriod)
What is the timer timeout length (in ms to two decimal places) of a PIC32MX's Timer 1 after running the following instructions given a bus clock frequency of 33.34 MHz?

T1CON = 0x00000020; → TCKPS bits are 10 (b/c Timer 1) → Prescale is 1:64
TMR1 = 0;
PR1 = 3649;
T1CONSET = 0x00008000;
Answer = **7.00**

1. Get the PS value from the T1CON value using the *7-4 Chart*. This one (above) is 1/64. **Make sure to use the appropriate chart like the other ones above. This is Timer1 (16-bit) but yours could be Timers2-5 (32-bit).**
2. Plug your PS value into this equation: $(PS/C_{BUS})*PR1$
3. This will give you the answer in seconds
   a. I find using the second PS number works
   b. (64)/(33.34M)*3649*10^3 = 7.0046

What is the timer timeout length (in micro seconds to one decimal place) of a PIC32MX's Timer 4 after running the following instructions given a bus clock frequency of 29.94 MHz? **Note that this is timer 4!**

T4CON = 0x00000050; -->TCKPS bits are 101, which is prescale 1:32
TMR4 = 0;
PR4 = 32;

T4CONSET = 0x00008000;
Answer = **34.2**

What is the timer timeout length (in ms to two decimal place) of a PIC32MX's Timer 1 after running the following instructions given a bus clock frequency of 7.33 MHz?

T1CON = 0x00000020;
TMR1 = 0;
PR1 = 3183;
T1CONSET = 0x00008000;
Answer = **27.79**

What is the timer timeout length (in micro seconds) of a PIC32MX's Timer 4 after running the following instructions given a bus clock frequency of 7.65 MHz?

T4CON = 0x00000020;
TMR4 = 0;
PR4 = 2104;
T4CONSET = 0x00008000;
Answer = **1100.0**

HW07-4(PIC32-Timers-TCKPSForFrequency)
Using a PIC32MX with a bus clock frequency of 16.90 MHz, produce a Timer 2 clock frequency as close to 0.085 MHz as possible with a single C instruction assuming PBCLK is used.
T2CONbits.TCKPS = 7;

1. Solve for prescale (PS) using equation from TimerClockFrequency question:
   a. $C_{BUS} = C_{CORE}$ / (PS)
   b. Or use this equation: PS = $C_{CORE}/(C_{BUS})$
   c. Result for above question: PS = 1:198.82
2. In the *7-4 Chart*, find the prescale value that is the ***next highest*** to the PS value you determined and get the bits needed to set that as the prescale value
   . Result, using above question: 1:256 → 111
3. In the command, make sure to use the appropriate timer number and make the command equal to the decimal result of that binary number you got in #2.
   . T2CONbits.TCKPS = *7*;

Using a PIC32MX with a bus clock frequency of 18.00 MHz, produce a Timer 5 clock frequency as close to 3.516 MHz as possible with a single C instruction assuming PBCLK is used.
T5CONbits.TCKPS = 2;

Using a PIC32MX with a bus clock frequency of 7.50 MHz, produce a Timer 2 clock frequency as close to 0.148 MHz as possible with a single C instruction assuming PBCLK is used.
T2CONbits.TCKPS = 6;

Using a PIC32MX with a bus clock frequency of 8.60 MHz, produce a Timer 3 clock frequency as close to 0.026 MHz as possible with a single C instruction assuming PBCLKis used.
T3CONbits.TCKPS = 7;

HW07-4(PIC32-Timers-TCKPSForPeriod)
Using a PIC32MX with a bus clock frequency of 14.90 MHz, produce a Timer 3 clock period as close to 0.403 micro seconds as possible with a single C instruction assuming PBCLK is used.
T3CONbits.TCKPS = 3;

Same thing as *TCKPSForFrequency* questions above, except you need to take the clock period and invert it, after which it is the same process as before.
$PS = C_{CORE}/(f_{CLK})$. This can also be done with $PS = C_{CORE} * t_{CLK}$. Choose the closest PS value that is closest to what is calculated.
A. (1/P) = target frequency
B. $C_{BUS} / x = F_{target}$
C. Choose closest prescale value to x (round up OR down)

Using a PIC32MX with a bus clock frequency of 10.80 MHz, produce a Timer 3 clock period as close to 9.156 micro seconds as possible with a single C instruction assuming PBCLK is used.
T3CONbits.TCKPS = 7;

Using a PIC32MX with a bus clock frequency of 5.10 MHz, produce a Timer 5 clock period as close to 26.102 micro seconds as possible with a single C instruction assuming PBCLK is used.
T5CONbits.TCKPS = 7;

Using a PIC32MX with a bus clock frequency of 8.60 MHz, produce a Timer 4 clock period as close to 0.148 micro seconds as possible with a single C instruction assuming PBCLK is used.
T4CONbits.TCKPS = 0;

Using a PIC32MX with a bus clock frequency of 7.50 MHz, produce a Timer 3 clock period as close to 1.323 micro seconds as possible with a single C instruction assuming PBCLK is used.
T3CONbits.TCKPS = 3;

Using a PIC32MX with a bus clock frequency of 13.60 MHz, produce a Timer 2 clock period as close to 1.459 micro seconds as possible with a single C instruction assuming PBCLK is used.
T2CONbits.TCKPS = 4;

HW07-4(PIC32-TxCON)
Give a single C instruction to configure a PIC32MX's Timer 4 with the following properties.

Timer is ENabled
Discontinue module operation when the device enters Idle Mode.
Gated time accumulation is DISabled.
1:1 prescale value.
32-bit Mode.
INternal peripheral Clock.
1. 1010000000**01**000
2. Hex: 0xA008

T4CON = 0xA008;

Instructions:
This is basically the PIC32-T1CON question, but for the other timers. Use the datasheet, page 157, for reference.
Build the binary number in this format:
A0**B**0,0000,**CDDD,E**0**F**0

A = ON
B = SIDL (1=discontinue in idle)
C = TGATE (gate time is enabled. Irrelevant if TC is on)
D = TCKPS (3bits) (0=1 scale, 1=2,2=4,3=8,4=16,5=32,6=64,7=256)
E = T32 (1=32 bit mode)
F = TCS (1=external clock)
Have fun!

Give a single C instruction to configure a PIC32MX's Timer 4 with the following properties.

Timer is ENabled
Continue module operation when the device enters Idle Mode.
Gated time accumulation is ENabled.
1:2 prescale value.
32-bit Mode.
INternal peripheral Clock.
T4CON = 0x8098;

Give a single C instruction to configure a PIC32MX's Timer 2 with the following properties.

Timer is ENabled
Discontinue module operation when the device enters Idle Mode.
Gated time accumulation is ENabled.
1:64 prescale value
32-bit Mode.
INternal peripheral Clock.
T2CON = 0xA0E8;

Give a single C instruction to configure a PIC32MX's Timer 4 with the following properties.

Timer is ENabled
Continue module operation when the device enters Idle Mode.
Gated time accumulation is ENabled.
1:32 prescale value.
32-bit Mode.
INternal peripheral Clock.
T4CON = 0x80D8

HW07-5(PIC32-ICx-WhatIsGraph)
Use the next two tables to check corresponding mode (ignore highlight) (from pg. 160)

bit 9 **FEDGE:** First Capture Edge Select bit (only used in mode 6, ICM=2:0= = 110)
1 = Capture rising edge first
0 = Capture falling edge first

bit 8 **C32:** 32-bit Capture Select bit
1 = 32-bit timer resource capture
0 = 16-bit timer resource capture

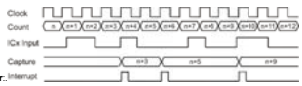
bit 7 **ICTMR:** Timer Select bit (Does not affect timer selection when C32 (ICxCON<8>) is set)
0 = Timer3 is the counter source for capture
1 = Timer2 is the counter source for capture

bit 6-5 **ICI<1:0>:** Interrupt Control bits
11 = Interrupt on every fourth capture event
10 = Interrupt on every third capture event
01 = Interrupt on every second capture event
00 = Interrupt on every capture event

Example 1
Consider the code below for a PIC32MX's Input Capture module:

IC1CONbits.ICM = 2;
IC1CONbits.ICI = 0;
IC1CONbits.FEDGE = 0;
IC1CONbits.ON = 1;

Which graph below correlates best to these settings?
1. "Simple capture event mode, every falling edge"
2. "Interrupt on every capture event"
3. "Capture on falling edge first"
4. "Module is enabled"

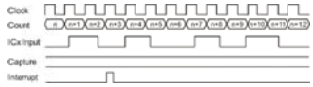


Answer:

Example 2
Consider the code below for a PIC32MX's Input Capture module:

IC1CONbits.ICM = 7;
IC1CONbits.ICI = 1;
IC1CONbits.FEDGE = 1;
IC1CONbits.ON = 1;

Which graph below correlates best to these settings?
1. "Interrupt-Only Mode"
2. "Interrupt on every second capture event"
3. "Capture on rising edge first"
4. "Module is enabled"

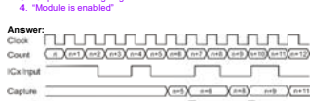


Example 3
Consider the code below for a PIC32MX's Input Capture module:

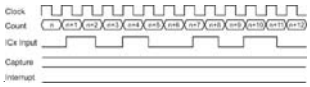
IC1CONbits.ICM = 6;
IC1CONbits.ICI = 1;
IC1CONbits.FEDGE = 1;
IC1CONbits.ON = 1;

Which graph below correlates best to these settings?
1. "Simple Capture Event mode – every edge, specified edge first and every edge thereafter"
2. "Interrupt on every second capture event"
3. "Capture on rising edge first"
4. "Module is enabled"

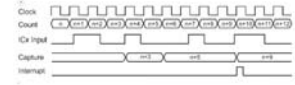
Answer:



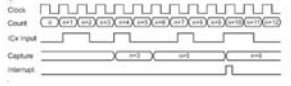
Example 4
Consider the code below for a PIC32MX's Input Capture module:
IC1CONbits.ICM = 0; "Input Capture module is disabled"
IC1CONbits.ICI = 2; "Interrupt on every capture event" → This doesn't matter b/c of ICM
IC1CONbits.FEDGE = 1; "Capture rising edge first" → Also doesn't matter b/c of ICM
IC1CONbits.ON = 1;
Which graph below correlates best to these settings?
Answer:



Example 5
Consider the code below for a PIC32MX's Input Capture module:
IC1CONbits.ICM = 2; "Simple Capture Event Mode-Every Falling Edge"
IC1CONbits.ICI = 2; "Interrupt on every third capture event"
IC1CONbits.FEDGE = 1; "Capture rising edge first"
IC1CONbits.ON = 1;
Which graph below correlates best to these settings?

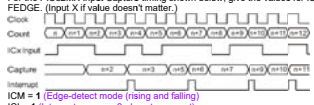


Example 6
Consider the code below for a PIC32MX's Input Capture module:
IC1CONbits.ICM = 2; "Simple Capture Event Mode-Every Falling Edge"
IC1CONbits.ICI = 2; "Interrupt on every third capture event"
IC1CONbits.FEDGE = 0; "Capture falling edge first"
IC1CONbits.ON = 1;
Which graph below correlates best to these settings?



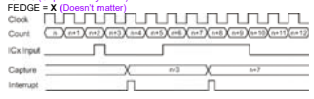
HW07-5(PIC32-ICx-WhatIsMode)
***NO SEMI-COLONS**

For the PIC32MX Input Capture timing shown below, give the values for ICM, ICI, and FEDGE. (Input X if value doesn't matter.)

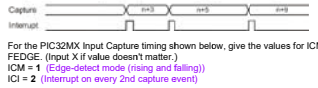


ICM = **1** (Edge-detect mode (rising and falling))
ICI = **1** (Interrupt on every 2nd capture event)
FEDGE = **X** (Doesn't matter due to ICM value)

For the PIC32MX Input Capture timing shown below, give the values for ICM, ICI, and FEDGE. (Input X if value doesn't matter.)
ICM = **3** (Simple capture event mode, every rising edge)
ICI = **0** (Capture every event)
FEDGE = **X** (Doesn't matter)

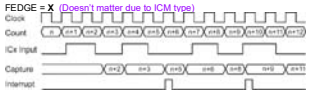


For the PIC32MX Input Capture timing shown below, give the values for ICM, ICI, and FEDGE. (Input X if value doesn't matter.)
ICM = **2** (Simple capture event mode, every falling edge)
ICI = **0** (Interrupt on every capture event)
FEDGE = **X** (Doesn't matter due to ICM type)



For the PIC32MX Input Capture timing shown below, give the values for ICM, ICI, and FEDGE. (Input X if value doesn't matter.)
ICM = **1** (Edge-detect mode (rising and falling))
ICI = **2** (Interrupt on every 2nd capture event)

FEDGE = **X** (Doesn't matter due to ICM type)



For the PIC32MX Input Capture timing shown below, give the values for ICM, ICI, and FEDGE. (Input X if value doesn't matter.)
ICM = **1** (Edge-detect mode (rising and falling))
ICI = **2** (Interrupt on every 2nd capture event)
FEDGE = **X** (Doesn't matter due to ICM type)

HW07-5(PIC32-ICxCON)

here for formatting. Delete when stuff is ready to be .
Instructions:
This is basically the PIC32-T1CON question, but for the Input Capture. Use the datasheet, page 160, for reference.
Build the binary number in this format (hyphens are placeholders for 0's): **A**-**B**---**CDEFFGHII**
A = ON
B = SIDL
C = FEDGE
D = C32
E = ICTMR
F = ICI
G = ICOV (if not mentioned, set equal to 0)
H = ICBNE (if not mentioned, set equal to 0)
I = ICM
Have fun!

Give a single C instruction to configure a PIC32MX's Input Capture Module 2 with the following properties:

Module is DISabled.
Discontinue module operation when in Idle Mode.
Capture Rising Edge First.
32-bit mode.
Timer 3 is counter source.
Interrupt on every fourth capture event.
Simple Capture Event Mode - every falling edge.
1. 0-1---1101100010
2. 0010001101100010
3. 0x2362

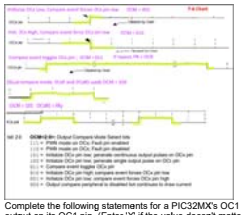
Answer: **IC2CON = 0x2362;**

Give a single C instruction to configure a PIC32MX's Input Capture Module 4 with the following properties:

Module is ENabled
Continue module operation when in Idle Mode.
Capture Rising Edge First.
32-bit mode.
Timer 3 is counter source.
Interrupt every capture event.
Edge Detect Mode.

Answer: **IC4CON = 0x8301;**

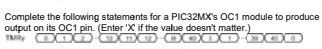
HW07-6(PIC32-OCx-WhatAreSettings)
.OCM command depends on graph below (from pg. 164)
PR2 is the highest value the TMRy value gets to before resetting to 0
OC1R is the value where the rising edge starts at (see graphic below)
OC1RS is the value of the *falling* edge of the first pulse. This only applies to diagrams that have pulses in them. If they don't, this is just X.



Complete the following statements for a PIC32MX's OC1 module to produce the given output on its OC1 pin. (Enter 'X' if the value doesn't matter.)

OC1CONbits.OCM = **001**;
PR2 = **78**;
OC1R = **1**;
OC1RS = **X**;

Complete the following statements for a PIC32MX's OC1 module to produce the given output on its OC1 pin. (Enter 'X' if the value doesn't matter.)



OC1CONbits.OCM = **010**;
PR2 = **40**;
OC1R = **39**;
OC1RS = **X**;

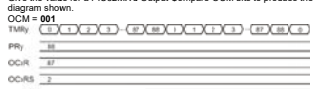
Complete the following statements for a PIC32MX's OC1 module to produce the given output on its OC1 pin. (Enter 'X' if the value doesn't matter.)

OC1CONbits.OCM = **101**;
PR2 = **62**;
OC1R = **52**;
OC1RS = **62**;

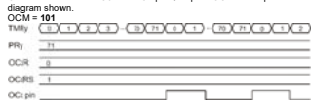
Complete the following statements for a PIC32MX's OC1 module to produce the given output on its OC1 pin. (Enter 'X' if the value doesn't matter.)

```
OC1CONbits.OCM = X ;
PR2 = 72 ;
OC1R = 72 ;
OC1RS = X ;
```

HW07-6(PIC32-OCx-WhatIsMode)
Give the value for a PIC32MX's Output Compare OCM bits to produce the timing diagram shown.
OCM = **001**



Give the value for a PIC32MX's Output Compare OCM bits to produce the timing diagram shown.
OCM = **101**



Give the value for a PIC32MX's Output Compare OCM bits to produce the timing diagram shown.
OCM = **011**

Give the value for a PIC32MX's Output Compare **OCM** bits to produce the timing diagram shown.
OCM = 100

HW07-6(PIC32-OCxCON)

This is the same process as the 7-5 ICxCON. Use pg. 164 for reference.
Give a single C instruction to configure a PIC32MX's Output Capture Module 4 with the following properties:

Module is ENabled
Continue module operation when in Idle Mode.
32-bit mode.
Timer 2 is clock source.
PWM mode on OCx. Fault pin DISabled.
OC4CON = 0x8026;

Give a single C instruction to configure a **PIC32MX's** Output Capture Module 2 with the following properties:

Module is DISabled.
Discontinue module operation when in Idle Mode.
32-bit mode.
Timer 2 is clock source.
Initialize OCx pin LOW. Event forces pin HIGH.
OC2CON = 0x2021;

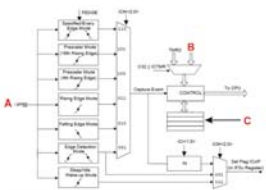
}
Answer = **222**
Example 5: (For incase you really need a lot of help like me.)
A PIC32MX with a bus clock frequency of 23.6 MHz has been set up with the following code for Timer 5:

```
void __ISR(_TIMER_5_VECTOR, ipl7) Timer_ISR;

T5CON = 0x00000060;
TMR5 = 0;
PR5 = 584;
T5CONSET = 0x000080000;
What does the value of C need to be in the ISR code below to get a square wave frequency as close to 14 Hz as possible?
void __ISR(_TIMER_5_VECTOR, ipl7) Timer_ISR;
{ static n=0;

if (++n >= C) {
LATAbits.LATA0 = ~LATAbits.LATA0;
n = 0;
}
IFS0bits.T5IF = 0;
}
```

Answer: 23.0

HW07-7(PIC32-Timer-CountForSquareWave-Period)
This is the same thing as CountForSquareWave-Frequency above.
Use the same formula, but note that they give you t_{SQR} directly this time.
$t_{SQR}=(C \times PR \times 2 \times n)/f_{BUS}$
A PIC32MX with a bus clock frequency of 23.9 MHz has been set up with the following code for Timer 4:

```
void __ISR(_TIMER_4_VECTOR, ipl7) Timer_ISR;


T4CON = 0x00000010; → (binary) 10000 → Prescale 1:2→ n=2 (TxCON table)
TMR4 = 0;
PR4 = 5;
T4CONSET = 0x000080000;
What does the value of C need to be in the ISR code below to get a square wave period as close to 1.5 ms as possible?
void __ISR(_TIMER_4_VECTOR, ipl7) Timer_ISR;
{ static n=0;
```



Which PIC32MX timers can have an asynchronous external clock? **Timer 1**
Which PIC32MX timers cannot be used for 32 bit timers? **Timer 1**
What is the lowest value possible in a PIC32MX's TMR1 register ? 0
What is the highest value possible in a PIC32MX's TMR5 register? **0xFFFF, FFFF, 65535**
What PIC32MX integrated peripheral counts clock pulses until an input signal changes state? **Input Capture**
What PIC32MX integrated peripheral works like a stopwatch? **Input Capture**

Chapter 8

HW08-1(WatchdoqTimers-General)
When is a watchdog a necessity in an embedded system?
A watchdog timer is a piece hardware that can be used to automatically detect software anomalies and reset the processor if any occur. Basically, A watchdog Timer is a counter that counts down to zero. However it's supposed to be reset every time the Program resets, if it isn't reset then it's assumed to be malfunctioning and the watchdog resets the program.

What happens if a watchdog chips timer reaches zero? **It restarts the Processor.**

What peripheral chip can be used to reset a microcontroller if it cannot get out of a loop structure?
Watchdog

Complete the following C instructions to produce a duty factor as close to 83.5 % as possible on a PIC32MX. 52 ÷ .835 - 1= 61.275

OC1R = 52;
OC1RS = 52;
PR2 = **61**;

Complete the following C instructions to produce a duty factor as close to 75.8 % as possible on a PIC32MX. 66 ÷ .758 - 1= 86.071

OC1R = 66;
OC1RS = 66;
PR2 = **86**;

HW09-2(PIC32-MaxPWMResolution-PR)
Equation: Max PWM Resolution Bits = $\log_2$(PRx)
This and the stuff below all (mostly) use this equation from the slides. I find it hard to understand/read quickly so I made the equation better.

Give the the maximum PWM resolution (in bits to two decimal places) of a PIC32MX PWM signal given a PBLCK of 19.5 MHz, and the following instructions.
T2CONbits.TCKPS = 0;
PR1 = 12;
Answer = **3.58** $\log_2((19.5e6 \times 12)/19.5e6)$

Give the the maximum PWM resolution (in bits to two decimal places) of a PIC32MX PWM signal given a PBLCK of 33.2 MHz, and the following instructions.
T2CONbits.TCKPS = 2;
PR1 = 386038;
Answer = **18.56**

Give the the maximum PWM resolution (in bits to two decimal places) of a PIC32MX PWM signal given a PBLCK of 37.7 MHz, and the following instructions.

T2CONbits.TCKPS = 1;
PR1 = 173775;
Answer = **17.41**

HW09-2(PIC32-MaxPWMResolution)
Equation: $\log_2((f_{PBCLK}/(f_{PWM} \times \text{prescale}))$? it works. Prescale comes from the binary number (Type B: TxCON chart)
On TI-89 you can do log base 2 by entering: $\log(f_{PBCLK}/(f_{PWM} \times \text{prescale}),2)$

HW07-7(Application-Shaft-MaxRPMs)
Consider a shaft which has 5 equally-spaced magnets placed around its circumference to trigger a hall-effect sensor connected to IC2 of a PIC32MX.

Assuming IC2 has been set up to capture every rising edge with a Timer 2 frequency of 26.4 kHz, what is the speed of the shaft (in RPMs) if the last value of TMR2 was 5991 and the new value of TMR2 is 20868, assuming no overflow has taken place?
dAnswer = **21.3**

Equation: $(t_{ticks} \times f_{TMR})/(n \times (\text{New-Old}))$
(60 * 26.4k) / (5*(20868 - 5991)
n = Amount of "ticks" per rotation. In this case it would be 5 b/c of the 5 magnets
t_{TMR} = Time you're measuring for in seconds. The question asks for the answer in RPM, so this 1 will be 1 minute, or 60 seconds.
New/Old = The values of the timer at the new and old measurement points

---

Consider a shaft which has 4 equally-spaced magnets placed around its circumference to trigger a hall-effect sensor connected to IC2 of a PIC32MX.

Assuming IC2 has been set up to capture every rising edge with a Timer 2 frequency of 4.5 MHz, what is the speed of the shaft (in RPMs) if the last value of TMR2 was 8396 and the new value of TMR2 is 49615, assuming no overflow has taken place?
Answer = **1638**

HW07-7(PIC32-Timer-CountForSquareWave-Frequency)
Get the prescale value using the 7-5 chart.
n = prescale value denominator: (e.g. 0x00000010 → (binary) 10000, which is prescale value 1:2, so n = 2)
t_{SQR}= 1 / f_{SQR} (Hz → s)
Equation: $f_{SQR}=(C \times PR \times 2 \times n)/f_{BUS}$
(f_{SQR} * f_{BUS}) / (PR x 2 x n) = C
Use that equation and solve for *C*, which is what question asks for.
*round up OR down

Example 1:
A PIC32MX with a bus clock frequency of 24.2 MHz has been set up with the following code for Timer 2:

```
void __ISR(_TIMER_2_VECTOR, ipl7) Timer_ISR;
T2CON = 0x00000030;
TMR2 = 0;
PR2 = 100;
T2CONSET = 0x000080000;
```

```
if (++n >= C)
{
LATAbits.LATA0 = ~LATAbits.LATA0;
n = 0;
}

IFS0bits.T4IF = 0;
}
```

$1.5 \times 10^{-3} = (C \times 5 \times 2 \times 2)/23.9 \times 10^{6}$. Solve for C. Why is n = 2? ***See addition in pink***
Answer = **1793**

HW07-7(PIC32-Timer-SquareWave-Frequency)
This time you're solving for f_{SQR}, which is from $f_{SQR}=1/t_{SQR}$, but you're still using the same equation: $f_{SQR}=(C \times PR \times 2 \times n)/f_{BUS}$
Example 1:
A PIC32MX with a bus clock frequency of 11.1 MHz has been set up with the following code for Timer 3:

```
void __ISR(_TIMER_3_VECTOR, ipl7) Timer_ISR;

T3CON = 0x00000040;
TMR3 = 0;
PR3 = 385;
T3CONSET = 0x000080000;
What is the frequency (in Hz to two decimal places) of the square wave generated on pin 0 of Port A generated with the ISR code?
void __ISR(_TIMER_3_VECTOR, ipl7) Timer_ISR;
{ static n=0;
if (++n >= 76)
{
LATAbits.LATA0 = ~LATAbits.LATA0;
n = 0;
}

IFS0bits.T3IF = 0;
}
```

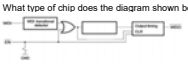
Answer = **11.85**

Example 2:
A PIC32MX with a bus clock frequency of 32.8 MHz has been set up with the following code for Timer 3:

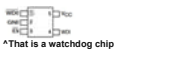
```
void __ISR(_TIMER_3_VECTOR, ipl7) Timer_ISR;
```

What peripheral chip can sometimes be used to reset a microcontroller if its supply voltage falls to an invalid value? **Watchdog**
What does a programmer have to do to prevent a Watchdog from timing out and resetting a processor?
It will have to be 'kicked' regularly, i.e. communicated with, once every given period.

What type of chip does the diagram shown below? **Watchdog Timer**



What is the purpose of the WDI pin? **watchdog input, likewise WDO is watchdog output**



^That is a watchdog chip

HW08-2(PIC32-InstructionsPerTimeout)
Equation: Instructions = $C_{SYSCLK} \times T_{TIMEOUT}$
Since all/most of these equations ask for the answer in the thousands, just use this:
Instructions = $C_{SYSCLK} \times T_{TIMEOUT} \times 10^{-3}$

How many typical instructions (in millions) could a PIC32MX with a SYSCLK of 2.8 MHz run before its watchdog times out given the following settings:
$(2.8*10^{6}Hz * 262.144s) = 734003200$ → Convert to millions (10^{6})→ 734.0032 (see slide 9 for table ~~edit: posted above)
WDTCONbits.WDTPS = 18;
734

How many typical instructions (in thousands) could a PIC32MX with a SYSCLK of 36.1 MHz run before its Watchdog times out given the following settings:
WDTCONbits.WDTPS = 2;
Answer = **144.0**

How many typical instructions (in thousands) could a PIC32MX with a SYSCLK of 33.9 MHz run before its Watchdog times out given the following settings:
WDTCONbits.WDTPS = 2;
Answer = **136.0**

Give the the maximum PWM resolution (in bits to two decimal places) of a PIC32MX PWM signal given a PWM frequency of 5.2 kHz, a PBLCK of 18.6 MHz, and the following instruction.
T2CONbits.TCKPS = 2;
Answer = **9.8**

Give the the maximum PWM resolution (in bits to two decimal places) of a PIC32MX PWM signal given a PWM frequency of 784 kHz, a PBLCK of 35.5 MHz, and the following instruction.
T2CONbits.TCKPS = 1;
Answer = **4.50**

Give the the maximum PWM resolution (in bits to two decimal places) of a PIC32MX PWM signal given a PWM frequency of 2.3 Hz, a PBLCK of 23.4 MHz, and the following instruction.
T2CONbits.TCKPS = 5;
Answer = **18.28**

HW09-2(PIC32-PWM-RMS)
Formula: $\sqrt{(OC1R)(PR1 + 1)} \times V_{PWM}$
Give the RMS voltage (to three decimal places) of a 4.5-Volt PWM signal on a PIC32MX with a PBLCLK of 22.9 MHz given the following settings:
PR1 = 66;
OC1R = 26;
OC1RS = 26;
T2CONbits.TCKPS = 7;
Answer = **2.803** V $\sqrt{(26/(66 + 1))} \times 4.5$

Give the RMS voltage (to three decimal places) of a 4.6-Volt PWM signal on a PIC32MX with a PBLCLK of 32.2 MHz given the following settings:
PR1 = 16;
OC1R = 7;
OC1RS = 7;
T2CONbits.TCKPS = 6;
Answer = **2.952** V

HW09-2(PIC32-PWMFrequency)
This requires 2 different pages of the datasheet.
A = PBDIV divider
B = TCKPS determined prescale value
Equation: $T_{PWM} = (PR + 1) \times (A / f_{SYSCLK}) \times (B)$
These questions are asking for the *frequency*, so you need to convert the output that is in time to that by doing $1/T_{PWM}$

What does the value of C need to be in the ISR code below to get a square wave frequency as close to 775 Hz as possible?

```
void __ISR(_TIMER_2_VECTOR, ipl7) Timer_ISR;
{ static n=0;

if (++n >= C) //the ++n means the value of n after it has been incremented
{
LATAbits.LATA0 = ~LATAbits.LATA0;
n = 0;
}

IFS0bits.T2IF = 0;
}
```

Answer = **20**
1/775 = (C*100*2*8) / (24.2M) , C = 19.512

Example 2:
A PIC32MX with a bus clock frequency of 30.5 MHz has been set up with the following code for Timer 4:

```
void __ISR(_TIMER_4_VECTOR, ipl7) Timer_ISR;


T4CON = 0x00000060;
TMR4 = 0;
PR4 = 39;
T4CONSET = 0x000080000;
What does the value of C need to be in the ISR code below to get a square wave frequency as close to 6 Hz as possible?
void __ISR(_TIMER_4_VECTOR, ipl7) Timer_ISR;
{ static n=0;

if (++n >= C)
{
LATAbits.LATA0 = ~LATAbits.LATA0;
n = 0;
}

IFS0bits.T4IF = 0;
}
```

Answer = **1018.0**

Example 3:

```
T3CON = 0x00000040;
TMR3 = 0;
PR3 = 7566;
T3CONSET = 0x000080000;
What is the frequency (in Hz to two decimal places) of the square wave generated on pin 0 of Port A generated with the ISR code?
void __ISR(_TIMER_3_VECTOR, ipl7) Timer_ISR;
{ static n=0;
if (++n >= 46)
{
LATAbits.LATA0 = ~LATAbits.LATA0;
n = 0;
}

IFS0bits.T3IF = 0;
}
```

Answer = **2.95**

HW07-7(PIC32-Timer-SquareWave-Period)
Same equation as the previous problems, but solving for period instead. This means you just need to find t_{SQR} without modifying (inverting) it.
$t_{SQR}=(C \times PR \times 2 \times n)/f_{BUS}$
A PIC32MX with a bus clock frequency of **25.3 MHz** has been set up with the following code for Timer 5:

```
void __ISR(_TIMER_5_VECTOR, ipl7) Timer_ISR;


T5CON = 0x00000050;
TMR5 = 0;
PR5 = 70;
T5CONSET = 0x000080000;
What is the period (in **micro seconds**) of the square wave generated on pin 0 of Port A generated with the ISR code?
void __ISR(_TIMER_5_VECTOR, ipl7) Timer_ISR;
{ static n=0;

if (++n >= 49)
{
LATAbits.LATA0 = ~LATAbits.LATA0;
n = 0;
```

HW08-2(PIC32-WatchdoqTimer-TimeoutPeriod)
LOOK AT THE CHART
Give the Time-out Period (in milliseconds) of a PIC32MX Watchdog Timer given the following instruction:

WDTCONbits.WDTPS = 3;
Answer = **8 ms**

Give the Time-out Period (in milliseconds) of a PIC32MX Watchdog Timer given the following instruction:
WDTCONbits.WDTPS = 21;
Answer = **1048576 ms**
The *Note 1* comment at the bottom of the chart is applied to this question

Give the Time-out Period (in milliseconds) of a PIC32MX Watchdog Timer given the following instruction:
WDTCONbits.WDTPS = 15;
32768 (slide 9 of 08-2…. Pasted above)
Give the Time-out Period (in milliseconds) of a PIC32MX Watchdog Timer given the following instruction:
WDTCONbits.WDTPS = 30;
1045876

HW08-2(PIC32-WatchdoqTimer-WDTPS)

Give a single C instruction to set the Time-out Period of a PIC32MX Watchdog Timer to as close to 1 min as possible.
WDTConbits.WDTPS = 16;

Give a single C instruction to set the Time-out Period of a PIC32MX Watchdog Timer to as close to 10 min as possible.
WDTConbits.WDTPS = 19;

Give a single C instruction to set the Time-out Period of a PIC32MX Watchdog Timer to as close to 1.2 ms as possible.
WDTConbits.WDTPS = 0;

Chapter 9

HW09-2(PIC32-DutyFactor-OCR)
Equation: OC1R = OC1RS = (PR2+1) x %
Complete the following C instructions to produce a duty factor as close to 59.9 % as possible on a PIC32MX. (84+1)*0.599=50.915

Give the frequency (in kHz to two decimal places) of a PIC32MX PWM waveform created by Timer 2 given the following C instructions, given that the SYSCLK is 3.7 MHz.
OSCCONbits.PBDIV = 1; → This would line up with "PBCLK is SYSCLK divided by 2" where 2 is A
T2CONbits.TCKPS = 0; → This means the prescale value is 1:1, so B = 1
PR2 = 955;
Answer = **1.94** (955+1) x (2/(3.7x10^{6})) x 1 → 5.167x10^{-4} → 1/5.167x10^{-4} → 1935.1 Hz → 1.94 kHz

Give the frequency (in Hz) of a PIC32MX PWM waveform created by Timer 2 given the following C instructions, given that the SYSCLK is 33.4 MHz.
OSCCONbits.PBDIV = 1;
T2CONbits.TCKPS = 2;
PR2 = 864;
Answer = **4827.0**

Give the freuency (in Hz to one decimal place) of a PIC32MX PWM waveform created by Timer 2 given the following C instructions, given that the SYSCLK is 17.3 MHz.
OSCCONbits.PBDIV = 0;
T2CONbits.TCKPS = 1;
PR2 = 27370;
Answer = **316.0**

HW09-2(PIC32-PWMPeriod)
Use the same equation used in the problem above, but don't convert it back to frequency. The equation, by default, gives the answer as a period (time), T_{PWM}.
Give the period (in milliseconds to three decimal places) of a PIC32MX PWM waveform created by Timer 2 given the following C instructions, given that the SYSCLK is 12.7 MHz.
OSCCONbits.PBDIV = 0;
T2CONbits.TCKPS = 1;
PR2 = 3523;
Answer = **0.555** ms

Give the period (in milliseconds to three decimal places) of a PIC32MX PWM waveform created by Timer 2 given the following C instructions, given that the SYSCLK is 27.2 MHz.
OSCCONbits.PBDIV = 2;
T2CONbits.TCKPS = 4;
PR2 = 566;
Answer = **1.334** ms

A PIC32MX with a bus clock frequency of 28.7 MHz has been set up with the following code for Timer 4:

```
void __ISR(_TIMER_4_VECTOR, ipl7) Timer_ISR;
T4CON = 0x00000050;
TMR4 = 0;
PR4 = 2;
T4CONSET = 0x000080000;
What does the value of C need to be in the ISR code below to get a square wave frequency as close to 11 Hz as possible?
void __ISR(_TIMER_4_VECTOR, ipl7) Timer_ISR;
{ static n=0;

if (++n >= C)
{
LATAbits.LATA0 = ~LATAbits.LATA0;
n = 0;
}

IFS0bits.T4IF = 0;
}
```

Answer = **20384.0**

Example 4:
A PIC32MX with a bus clock frequency of 35.1 MHz has been set up with the following code for Timer 5:

```
void __ISR(_TIMER_5_VECTOR, ipl7) Timer_ISR;
T5CON = 0x00000010;
TMR5 = 0;
PR5 = 4;
T5CONSET = 0x000080000;
What does the value of C need to be in the ISR code below to get a square wave frequency as close to 9.9 kHz as possible?
void __ISR(_TIMER_5_VECTOR, ipl7) Timer_ISR;
{ static n=0;

if (++n >= C)
{
LATAbits.LATA0 = ~LATAbits.LATA0;
n = 0;
}

IFS0bits.T5IF = 0;

}

IFS0bits.T5IF = 0;
}
```

Answer = **8677**

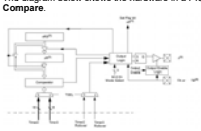
HW07(PIC32-Timers-General)
What signal values/transitions can a PIC32MX's output compare module produce at a given pin?
Toggle, Low Value, High Value, Low-High-Low Pulse, and Continuous Pulses

What PIC32MX integrated peripheral works like a "Alarm Clock?"
Output Compare

What do you call a timer which allows an external signal to control when a timer counts and when it doesn't?
Gated Timer

What is register in a PIC32MX contains the timer count values? **TMR**
Which register in a PIC32MX contains the value a timer counts up to? **PR or PRx**

The diagram below shows the hardware in a PIC32MX used to perform a\[n] **Output Compare**.



For the PIC32MX module diagram below, to what does Label B refer?
Answer = **TMR3**
C is the FIFO Buffer
A is the ICx
Found on pg.1 of 7-5 notes

OC1R = **51**;
OC1RS = **51**;
PR2 = 84;

Complete the following C instructions to produce a duty factor as close to 26.0% as possible on a PIC32MX. (86+1)*0.260=22.62
OC1R = **23**;
OC1RS = **23**;
PR2 = 86;

Complete the following C instructions to produce a duty factor as close to 78.4% as possible on a PIC32MX. (50+1)*0.784 = 39.984
OC1R = **40**;
OC1RS = **40**;
PR2 = 50;

HW09-2(PIC32-DutyFactor-PR)
(OC1R+DF)-1 = PR ~~***Round the result***~~
Complete the following C instructions to produce a duty factor as close to 16.4 % as possible on a PIC32MX. 63+0.164=384.146 → 383

OC1R = 63;
OC1RS = 63;
PR2 = **383**;

Complete the following C instructions to produce a duty factor as close to 1.0 % as possible on a PIC32MX. 43+0.01-1 = 4299
OC1R = 43;
OC1RS = 43;
PR2 = **4299**;

Complete the following C instructions to produce a duty factor as close to 70.3 % as possible on a PIC32MX. 44+0.703-1=61.589
OC1R = 44;
OC1RS = 44;
PR2 = **62**;

Complete the following C instructions to produce a duty factor as close to 13.1 % as possible on a PIC32MX. 47 + .131 - 1= 357.78

OC1R = 47;
OC1RS = 47;
PR2 = **358**;

HW09-2(PIC32-PWMs-General)

What does PWM stand for?
Pulse Width Modulation
What property/characteristic of a microcontroller allows a PWM to be useful?
PWM allows a microcontroller to interface with many analog components. The microcontroller, being a digital device, only outputs a binary result, so PWM allows some components to be "tricked" by sending pulses and a specific frequency.
Give an example of a practical use of a PWM waveform.
Controlling a motor's speed
How do you get a 100% duty factor using the PWM of a PIC32MX?
By having OCxR greater than PRy
What does "Modulation" mean in PWM?
A technique used in communication systems for coding the height of the signal. So it controls the amplitude of the signals.
What is the maximum number of duty factors that a 16-bit Output Compare module on a PIC32MX can produce?
65536
What is the maximum number of PWM that can be running at any given time on a PIC32MX?
5
What is the maximum number of different PWM periods that a 32-bit Output Compare module can produce on a PIC32MX?
4294967296
How do you get a 100% duty factor using the PWM of a PIC32MX?
When OCxR is loaded with a 0x0000, the OCx pin will remain low giving a 0% duty cycle.
What is the maxium number of duty factors that a 32-bit Output Compare module on a PIC32MX can produce?
2^32
What is the purpose/use of PWM "fault protection?"
To prevent a steady high from being left on the PWM control line in case of a reset/power up, etc.
What is the maximum number of PWM that can be running at any given time on a PIC32MX?
5
What is the maximum number of duty factors that a 16-bit Output Compare module on a PIC32MX can produce?
2^16 = 65536

ECE 3710 TEST 3

Does anyone know the answer to these questions?
What is the minimum sampling time in the sample-convert sequence when using a PIC32MX's ADC?

Because acquisition has a minimum time of 132ns and the clock period of the ADC is at least 65ns while conversion takes 12 TAD, the minimum sample time = (65ns * 12 TAD) + 132ns = 912ns.
Thank you. That's what I thought, but the notes were not very clear.

Are we sure this is right?

Is there a minimum conversion time in the sample-convert sequence when using a PIC32MX's ADC? If so, what is it?

Found this in the ADC datasheet:

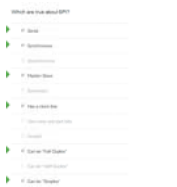
Param No.	Symbol	Characteristics	Min.	Typical(1)	Max.	Units	Conditions
Clock Parameters							
AD50	TAD	ADC Clock Period(3)	65	—	—	ns	See Table 29-33
Conversion Rate							
AD55	TCONV	Conversion Time	—	12 TAD	—	—	—
AD56	FCNV	Throughput Rate (Sampling Speed)	—	—	1000	ksps	AVDD = 3.3V to 3.6V
			—	—	400	ksps	AVDD = 2.5V to 3.6V
AD57	TSAMP	Sample Time	1 TAD	—	—	—	TSAMP must be ≥ 132 ns

So technically there isn't a minimum conversion time, but the typical conversion time is 12 TAD.

Chapters 10-20 (no 14 - 18)
Homework Equations and Explanations
Explanations and equations are in purple.
*when typing your answers please don't type word for word
***All free response questions are unconfirmed, take them with a bag of rice
If you have a question with a picture, you can save the picture and the default file name contains the answer usually (For HW only)

Chapter 10

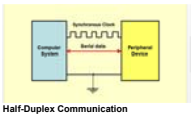
HW10-1(SynchronousSerialComm-General)
What type of Serial Communication is shown below?



Give the name of all SPI signal lines / SPI has the following signals.

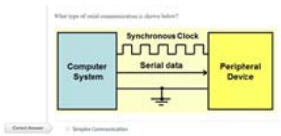
MOSI, MISO, SS#, SCK

HW10-3(SPI-PIC32-Clock)
Helpful Chart



Half-Duplex Communication

What type of serial Communication has two serial data lines each allowing communication in one direction?
Full Duplex Communication



What makes Synchronous Serial Communication "Synchronous"?
There is a protocol where each device communicating uses a common clock to synchronize data transmission.

What is Half-Duplex communication?
A system through which each party can communicate with each other, but not at the same time.
Example: Walkie-Talkie

What is Duplex communication?
Duplex, same as full duplex, is a form of communication through which both parties can send and receive data at the same time.

NOTE:
- Full-Duplex allows for communication in either direction at any time (2 lines). An example is HTTP Pipelining.
- Half-Duplex allows for communication in one direction at one time then can switch to the opposite direction (one line two arrows. An example is the POP3 Protocol (as well as FTP, NNTP, SMTP).
- Simplex allows for communication in one direction only and doesn't allow switching directions. This is sometimes specified programmatically as Simplex Server or Simplex Client.

Which of the following things must a sender and receiver agree upon before they can communicate in a synchronous serial fashion?

The speed of the clock, the order of the bits (MSb or LSb first), Number of bits per symbol, Simplex or Duplex communication.

Give an example of synchronous serial communication protocols mentioned in the notes.

What type of serial communication has a single serial data lines allowing communication in both directions? **Half-Duplex Communication**

HW10-2(SPI-General)
What does MISO stand for?
Master In-Slave Out

What function does the 74HC595 chip perform?
Serial-to-Parallel Conversion

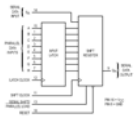
What does SPI Stand for?
Serial Peripheral Interface

SPI has the following signals.
MOSI, MISO, SS#, SCK

Identify the 7400-series chip shown.

74HC595

Identify the 7400-series chip shown.



74HC597

What does the SPI **SS#** line do?
The slave select line is normally low active and once activated lets the slave know it is about to be sent data. Normally each slave has a different slave select line.

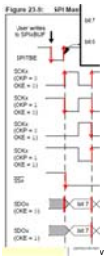
SPI is a _____ protocol.
Master-slave synchronous serial

What's the purpose of a **74HC597** chip when using a **SPI** Port?
This chip is a shift register, so it can be configured to take a serial input from the SPI port and convert it to a parallel output.
Maybe I'm dumb but on his scripts it has the following (which I think is different):
2 The 74H595 IC is a serial-to-parallel chip which takes in a byte of serial data and then drives its 8 outputs with the data read in.
3 The 74H597 does the opposite, taking in a parallel data byte, and then transmitting that byte serially at its output.

Give the name of all SPI signal lines / SPI has the following signals.

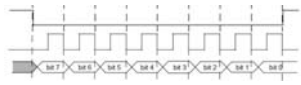
MOSI, MISO, SS#, SCK

HW10-3(SPI-PIC32-Clock)
Helpful Chart



Which graph represents a PIC32MX SPI timing with where **CKP = 1** and **CKE = 0**?

Which graph represents PIC32MX SPI timing with where **CKP=0** and **CKE=1?**



HW10-3(SPI-PIC32-SPIxCON) - **the table from the manual works too (pg 166--168)**

A00B 0CCC D000 00EF G0HI JJ0K LM0N OOPP
A. FRMEN Frame SPI support (1=enabled)
B. MSSEN: Master Mode Slave Select Enable (1=slave select SPI enabled)
C. FRMCNT: Frame Sync Pulse Counter (0=1 chars, 5 [max]=32 chars, x=2^n chars)
D. MCLKSEL: Master Clock Enable (1=BRG uses REFCLK,0=BRG uses PBCLK)
E. SPIFE: Frame Sync Pulse Edge Select (1=FSP coincides with first bit clock, 0=precedes)
F. ENHBUF: Enhanced Buffer Enable (1=enabled)
G. ON: SPI Peripheral on (1=enabled)
H. SIDL: Stop In Idle Mode (1=discontinue in idle mode)
I. DISSDO: Disable SDOx pin (1=SDOx pin not used by module-controlled by PORT)
J. MODE: if auden=0->0=8 bit mode, 1=16 bit mode, 2/3=32 bit mode
K. CKE: SPI Clock Edge Select (1=serial output data changes when from active to idle)
L. SSEN: Slave Select Enable (1=SSx pin used for slave mode)
M. CKP: Clock Polarity Select (1=idle state for clock is high & active is low)
N. DISSDI: Disable SDI (1=SDI not used by SPI-used by PORT)
O. STXISEL: SPI Transmit Buffer Empty INT(3=not full,2=½ empty,1=empty,0=when done)
P. SRXISEL: SPI Receive Buffer Full INT (3=full,2=½ full,1=not empty,0=last word read

Give a Single C instruction to configure a **PIC32MX's** SPI Module 2 with the following properties:
-Slave Select Support Enabled
-Enhanced buffer mode enabled
-16-bit Mode
-Receive is generated when the buffer is NOT empty.
Answer: SPI2CON = 0x10010401;

Give a Single C instruction to configure a **PIC32MX's** SPI Module 2 with the following properties:
-Frame Support Enabled
-REFCLK is used for Baud Rate
-Frame Synchronization pulse coincides with first by clock
-Discontinue Module operation when the device enters Idle Mode
-Receive is generated when the buffer is NOT EMPTY
Answer: SPI2CON = 0x80822001;

Give a Single C instruction to configure a **PIC32MX's** SPI Module 2 with the following properties:
-Slave Select Support Enabled
-Frame Synchronization pulse Coincides with first Bit clock.
-16-bit Mode
-Serial output data changes on transition from active clock state to idle clock state
-Receive is generated when the buffer is NOT EMPTY
Answer: SPI2CON = 0x10020501;

Give a single C instruction to configure a **PIC32MX's** SPI Module 2 with the following properties:
- Slave select support ENabled.
- Frame synchronization pulse coincides with first bit clock.
- Discontinue module operation when the device enters Idle Mode.
- Idle state for clock is a HIGH level.
- Receive is generated when the buffer is NOT EMPTY.
(Assume all parameters not mentioned are 0's).
SPI2CON = 0x10022041;

Give a single C instruction to configure a PIC32MX's SPI Module 1 with the following properties:
- Frame support ENabled.
- Generate a frame sync pulse on every 16 data characters.
- REFCLK is used for Baud Rate.
- SDOx pin is NOT controlled by module.
- Receive is generated when the buffer is at least HALF FULL.
Answer: SPI1CON = 0x84801002;

Give a single C instruction to configure a PIC32MX's SPI Module 1 with the following properties:

- Slave select support ENabled.
- Frame synchronization pulse coincides with first bit clock.
- Discontinue module operation when the device enters Idle Mode.
- Serial output data changes on transition from active clock state to idle clock state.
- Receive is generated when the buffer is NOT EMPTY.
Answer: SPI1CON = 0x10022101;

HW10-3(SPI-PIC32-SPIxCON2)

NOTE: The data sheet may have a typo b/c it doesn't show bit 4, just enter 0 for bit 4.
Give a single C instruction to configure a PIC32MX's SPI Module 2 with the following properties:

- Data from RX FIFO is sign extended.
- Audio Protocol ENabled.
- Audio Data is Mono.
- PCM/DSP Mode
SPI2CON2 = 0x808B;

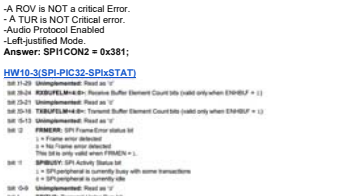
Give a single C instruction to configure a PIC32MX's SPI Module 2 with the following properties:

- A ROV is NOT a critical error.
- Audio Protocol ENabled.
- Audio Data is Mono.
- PCM/DSP Mode
SPI2CON2 = 0x28B;

Give a single C instruction to configure a **PIC32MX's** SPI Module 1 with the following properties:

-A ROV is NOT a critical Error.
- A TUR is NOT Critical error.
-Audio Protocol Enabled
-Left-justified Mode.
Answer: SPI1CON2 = 0x381;

HW10-3(SPI-PIC32-SPIxSTAT)



Given the value of a PIC32MX's **SPIxSTAT** register shown below,
SPI1STAT = 0x4000923;
is the following statement true or false?
SPI peripheral is busy?
TRUE
look at bit 11

Given the value of a PIC32MX's **SPIxSTAT** register shown below,
SPI2STAT = 0x31B11A9;
is the following statement true or false?

Transmit Buffer Element Count Bits are NOT equal to 27.
FALSE

```
SPI2STAT = 0x81318C1;
```

is the following statement true or false?
Transmit Buffer IS empty. **FALSE**

HW10-3(SPI-PIC32-WhatIsBaudRate)
Baud Rate = (f_{PBCLK})/(2*(SPIxBRG+1)) (note the units)

A PIC32MX has a PBCLK frequency of 32.8 Mhz. What is the SPI1 Baud Rate (in kb/s to one decimal place) given the following register settings?
SPIxBRG = 473;
Answer = 34.6

--
A PIC32MX has a PBCLK frequency of 30.3 MHz. What is the SPI1 Baud Rate (in kb/s to one decimal place) given the following register settings?

SPI1BRG = 465;
Answer = 32.5

A PIC32MX has a PBCLK frequency of 33.5 MHz. What is the SPI1 Baud Rate (in kb/s to one decimal place) given the following register settings?

SPI1BRG = 0x071;
Answer = 146.9

HW10-3(SPI-PIC32-WhatIsSPIxBR)
Given a PIC32MX PBCLK value of **29.5 MHz**, give the value of **SPIxBRG** to produce a Baud Rate as close to **216.0 kb/s** as possible.
baud rate = ($f_{frequencyPBCLK}$)/(2* (SPIxBRG +1))
SPIxBRG = ($f_{frequencyPBCLK}$)/(2*baud rate) - 1
216.0*10^3 b/s = (29.5*10^6 Hz)/(2*(SPIxBRG +1))
SPIxBRG = 67

Given a PIC32MX PBCLK value of 14.2 MHz, give the value of SPIxBRG to produce a Baud Rate as close to 31.6 kb/s as possible.
SPIxBRG = **224**

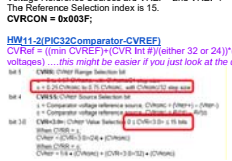
Chapter 11

HW11-2(PIC32-CVRCON)

Give a single C instruction to configure the PIC32MX comparator Modules with the following Properties:
Module is Enabled
Voltage level is output to pin.
Voltage Range is 0.25 CVRSRC to 0.75 CVRSRC.
Voltage Reference Sources are VREF+ and VREF-.
The Reference Selection index is 12. This is the last four bits of the binary value
CVRCON = 0x805C;
Give a single C instruction to configure the PIC32MX comparator Modules with the following Properties:
Module is Enabled
Voltage level is disconnected from output pin
Voltage Range is 0.25 CVRSRC to 0.75 CVRSRC.

Voltage Reference Sources are VREF+ and VREF-.
The reference selection index is 11.
CVRCON = 0x801B;
Give a single C instruction to configure the PIC32MX comparator Modules with the following Properties:
Module is Disabled.
Voltage level is disconnected from output pin.
Voltage Range is 0 to .67 CVRSRC.
Voltage Reference Sources are VREF+ and VREF-.
The Reference Selection index is 15.
CVRCON = 0x003F;

HW11-2(PIC32Comparator-CVREF)
CVRef = {(min CVREF)+(CVR Int #)/(either 32 or 24)*(Difference between CVRR voltages) ….*this might be easier if you just look at the datasheet?*



Given the following C instructions, what is the value of CVRef for the PIC32 comparator (to three decimal places) given that VREF+ = 3.13, VREF- = 0.58, and AVDD = 2.85, AVSS = 1.35?
CVRCONbits.CVRR = 1;
CVRCONbits.CVRSS = 1;
CVRCONbits.CVR = 2;
Answer = 0.212

Given the following C instructions, what is the value of CVRef for the PIC32 comparator (to three decimal places) given that VREF+ = 2.75, VREF- = 0.31, and AVDD = 2.43, AVSS = 0.99?

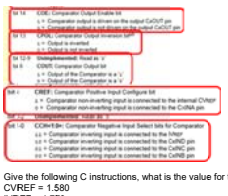
CVRCONbits.CVRR = 1;
CVRCONbits.CVRSS = 0;
CVRCONbits.CVR = 14;
Answer = 0.84
14/24*(2.43-.99) = 0.84

Given the following C instructions, what is the value of CVRef for the PIC32 comparator (to three decimal places) given that VREF+ = 2.06, VREF- = 1.16, and AVDD = 2.19, AVSS = 0.35?

```
CVRCONbits.CVRR = 0;
CVRCONbits.CVRSS = 1;
CVRCONbits.CVR = 4;
```
Answer 0.338
(2.06-1.16)/4*(2.06-1.16)*4/32

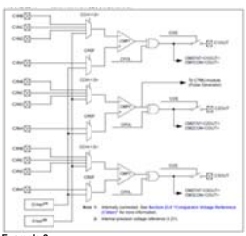
HW11-2(PIC32Comparator-Output)
Just look at the picture below
and reason through it.
if INV>NONInv, output = 0
The output is zero by default
and if there are not ones being
outputted it is a tri state (when COE = 0).

CREF voltage < CCH voltage => 0 if CPOL = 0, 1 if CPOL = 1
CREF voltage > CCH voltage => 1 if CPOL = 0, 0 if CPOL = 1
COE = 0 => HiZ



Give the following C instructions, what is the value for the PIC32 comparator given that
CVREF = 1.580
IVREF = 1.770
C1INA = 2.000
C1INB = 2.481
C1INC = 0.114
C1IND = 0.841
CM1CONbits.COE = **1**;
CM1CONbits.CPOL = 0;
CM1CONbits.CREF = 0;

CM1CONbits.CCH = **1**;



Example 2
Given the following C instructions, what is the value for the PIC32 comparator given that

CVREF = 2.690
IVREF = 1.792
C2INA = 1.770
C2INB = 2.543
C2INC = 0.961
C2IND = 0.182

CM2CONbits.COE = **1**;
CM2CONbits.CPOL = 1;
CM2CONbits.CREF = 0;
CM2CONbits.CCH = 2;
COUT = 0

Example 3
Given the following C instructions, what is the value for the PIC32 comparator given that

CVREF = 2.920
IVREF = 0.164
C2INA = 1.280
C2INB = 0.905
C2INC = 1.720
C2IND = 2.549

CM2CONbits.COE = 0;
CM2CONbits.CPOL = 1;
CM2CONbits.CREF = 1;
CM2CONbits.CCH = 2;
COUT = HiZ

Example 3
Given the following C instructions, what is the value for the PIC32 comparator given that

CVREF = 2.720
IVREF = 1.696
C1INA = 1.540
C1INB = 0.963F
C1INC = 0.126
C1IND = 2.641

CM1CONbits.COE = 1;
CM1CONbits.CPOL = 0;
CM1CONbits.CREF = 0;
CM1CONbits.CCH = 3;
COUT = 1 **Shouldn't this be 0?**

Chapter 12
HW12-1(DACs-Bits)
DACbits = log_2((High Voltage - Low Voltage) / (Increment Voltage)) (Always round up to the next highest integer. Never Round Down.

If we need to produce voltages from 0 to 10 Volts, in increments of 2 mV, what's the fewest number of bits that our DAC input must have?
DACbits = log((High Voltage - Low Voltage)/Increment Voltage)/log(2) (Round up)
log((10-0)/(2*10^-3))/log(2) = **13** Bits

HW12-1(DACs-General)

Every answer above can be found here:

https://www.maximintegrated.com/en/app-notes/index.mvp/id/641

What is a “relative accuracy error” as it pertains to a DAC?
It’s the Output error between the measured response and a line running from the output given an input of -0, and the output given an input of $2^N – 1$.
Give an Example of how a **DAC** can be used in an embedded system.
They could be used to generate an audio signal from 0’s and 1’s to an analog signal that could be sent to a handphone amp.
what is a “zero order hold”?
A zero order hold is used in the reconstruction of an analog signal for a DAC. Basically, Its used to describe the effect of converting a discrete time signal into a continuous time signal.
What does "unipolar" mean as it pertains to a DAC?
The binary input can only be a positive number.

What is a "full scale error" as it pertains to a DAC?
The difference between the actual value that triggers the transition to full-scale and the ideal analog full-scale transition value.

What is a the "zero error" of a DAC?
It indicates how well the actual transfer function matches the ideal transfer function at a single point.

What does “offset binary” mean as it pertains to a DAC?

Why are you not generally going to find a **DAC** on a microcontroller?"

HW12-1(DACs-Levels)
*** For the following, there are two equations based on whether the DAC is unipolar or bipolar
*** And pay attention to whether they want the answer in V or mV
If a 11-bit unipolar DAC has a reference voltage of 6.7 Volts, what is the smallest non-zero voltage (in mV) that can be produced?
3.271
Unipolar Equation: SmallestVoltage = RefVolt / (2^Bits)

If a 10-bit bipolar DAC has a reference voltage of 10.8 V, what is the smallest positive voltage (in mV to three decimal places) that can be produced?
21.094
Bipolar Equation: SmallestVoltage = RefVolt / (2^(Bits - 1))

If a 12-bit bipolar DAC has a reference voltage of 13.3 Volts, what is the **maximum** voltage (in Volts to four decimal places) that can be produced?
13.2935
Bipolar Equation: MaxVoltage = RefVolt * [2^(Bits-1) - 1] / 2^(Bits-1)

If a 10-bit unipolar DAC has a reference voltage of 12.8 Volts, what is the maximum voltage (in Volts to four decimal places) that can be produced?
12.7875
Unipolar Equation: MaxVoltage = RefVolt * [2^(Bits) - 1] / 2^(Bits)

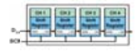
HW12-1(DACs-Values-Offset)
Given a 10-bit Offset-Binary bipolar DAC with a Vref of 10.0 V, what voltage would be output (to 3 decimal places) for an input code of 01 0010 0001?
SmallestVoltage = [(Vref / ($2^{(Bits - 1)}$)) * Binary]-Vref
((10/(2^(10-1)))*289)-10 = **-4.355** V

HW12-1(DACs-Values)
***Again, watch for whether the DAC is unipolar (use bits) or bipolar (use bits-1)
Given a 10-bit unipolar DAC with a Vref of 10.0V, what voltage would be output (to 3 decimal places) for an input code 0f 01 1000 0001?
3.760
Unipolar Equation: Voltage = [Vref / (2^bits)] * (binary code converted to decimal)
= (10 / 2^10) * (385)
= 3.760

Bipolar Equation: Voltage = (Vref / 2^(bits - 1)) * (binary code converted to decimal)

HW12-1(MultichannelDAC)
EXAMPLE SET 1
What voltage (to three decimal places) would come out of Channel 3 of the 6-bit DAC below given Vref = 3 V if the following bits are sent and the right-most bits are transmitted first.
0111 0001 0100 0101 0001 1010
Vout = (Vref*D_{in}) / 2^(#ofBits)
D_{in} = Decimal value of (# of Bits) in specified channel
Always start counting from the right. Pay attention to MSB (Most significant bit) and LSB (Least significant bit),
(3*20)/(2^6) = **0.938** V

What voltage (to three decimal places) would come out of **Channel 1** of the **7-bit** DAC below given Vref = **3 V** if the following bits are sent and the right-most bits are transmitted first.
1110 0101 0000 1000 0110 1001 0000



2.672
What voltage (to three decimal places) would come out of Channel 1 of the 5-bit DAC below given Vref = 3 V if the following bits are sent and the right-most bits are transmitted first.
0101 0011 1010 0100 0110 1010 0110

0.563

EXAMPLE SET 2
What voltage (to three decimal places) would come out of Channel 4 of the 5-bit DAC below given Vref = 3 V if the following bits are sent and the right-most bits are transmitted first.
0110 1110 1011 0000 1101

1.219

What voltage (to three decimal places) would come out of Channel 4 of the 7-bit DAC below given Vref = 5 V if the following bits are sent and the right-most bits are transmitted first.
1011 1100 0111 0110 0111 0010 1111 0001 1101

3.672

What voltage (to three decimal places) would come out of Channel 3 of the 5-bit DAC below given Vref = 5 V if the following bits are sent and the right-most bits are transmitted first.
1001 1111 0101 1001 0110 1011 0101 1001

2.813 → This one doesn’t work as expected

HW12-2(ADCBits)
ADCbits = log_2((High Voltage - Low Voltage) / Resolution Voltage) (Always round up to the next highest integer. ***Never NEVER NEVER*** Round Down.

If we need to sample voltages from 0 to 11.4 Volts, with a resolution of 62 mV, what's the fewest number of bits that our ADC output must have?
ADCbits = log((High Voltage -Low Voltage)/Resolution Voltage)/log(2) (Round up)
log((11.4-0)/(62*10^-3))/log(2) = **8**

HW12-2(ADCLevels)
*** For the following, there are two equations based on whether the DAC is unipolar or bipolar
*** And pay attention to whether they want the answer in V or mV
If a 10-bit unipolar ADC has a reference voltage of 12.5 Volts, what voltage (in mV) does an output value of 1 represent?
12.207
Unipolar Equation: Voltage = Vref / 2^bits

If a 13-bit bipolar ADC has a reference voltage of 5.3 Volts, what voltage (in V to three decimal places) does an output of 1 represent?
-5.299
Bipolar Equation: Voltage = (Vref / 2^(bits-1)) - Vref

If a 12-bit bipolar ADC has a reference voltage of 13.9 Volts, what voltage (in V to four decimal places) does an output code value of all 1's represent?
Bipolar Equation: Voltage = (Vref / 2^(bits-1)) - Vref
All 1’s -> make positive
13.8932

If a 10-bit unipolar ADC has a reference voltage of 4.4 Volts, what voltage (in V to four decimal places) does an output code value of all 1's represent?
Bipolar Equation worked here for some reason
4.3957

HW12-3(PIC32-ADC-WhatIsCode-Decimal-16bit)

*Leading 1: -512 (signed)
*Leading 0: +512 (signed)

If form=0, find decimal of last 10 bits

If form=1, 10th bit is sign (every bit before it should be the same) and last 9 bits are the decimal (signed int), remember 0x200=-512 and then you add the next 9 bits to -512 if 10th bit is 1

If form=2, first 10 bits are decimal and answer = decimal / (2^n) (n is always 10, so 1024)

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 1;
what decimal value does the following binary code sequence represent?
1111 1110 1101 1110
-290.0 -512+0b11011110

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 0;
what decimal value does the following binary code sequence represent?
0000 0001 0110 0001
353.0

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 2;
what decimal value does the following binary code sequence represent?
0110 0011 1000 0000
0.389

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 3;
what decimal value does the following binary code sequence represent?
1001 1000 1100 0000
-0.403
FORM = 3 means it’s a Signed Fractional 16-bit, so you disregard the last six 0’s
The first bit is the sign, so you start with -2^9 = -512
You then convert the next 9 binary numbers to decimal
001100011 = 99
-512 + 99 = -413
Answer = decimal / 2^n (n is always 10 so 1024)
-413 / 1024 = -0.403

HW12-3(PIC32-ADC-WhatIsCode-Decimal-32bit)
If form=5, bit 9 is sign and every bit before it should be the same, 0xFFFFFF000=-512
If form=6, first 10 bits are decimal, answer=number/(2^n), n=10
If form=7, 1st bit is sign, next 9 are decimal, answer=signed int/(2^n), n=10

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 5;
what decimal value does the following binary code sequence represent?
1111 1111 1111 1111 1111 1111 1111 0100
-12.0

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 7;
what decimal value does the following binary code sequence represent?
0111 1100 0000 0000 0000 0000 0000 0000
0.484

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 7;
what decimal value does the following binary code sequence represent?
1111 0110 1000 0000 0000 0000 0000 0000
-0.037 (-512+0b111011010)/1024

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 5;
what decimal value does the following binary code sequence represent?
0000 0000 0000 0000 0000 0001 0100 1001
329.0 0b111011010 → dec (because sign bit is 0)

HW12-3(PIC32-ADCVoltage-Fractional-16bit)

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 2;
AD1CON2bits.VCFG = 3;
and has the following Reference voltages:
AVDD = 4.2 Volts.
AVSS = 0 Volts.
VREF+ = 2.6 Volts.
VREF- = 0 Volts.
What voltage value (to three decimal places) does the following binary sequence represent?
1001 1110 1000 0000
1.61 0b1001111010/1024*2.6

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 6;
AD1CON2bits.VCFG = 3;
and has the following Reference voltages:
AVDD = 2.4 Volts.
AVSS = 0 Volts.
VREF+ = 2.1 Volts.
VREF- = 0 Volts.
What voltage (to three decimal places) value does the following binary sequence represent?
1011 1010 0000 0000 0000 0000 0000 0000
1.744 0b1011101000/1024*2.4 ...but this doesn’t make sense?...ya idk wtf

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 6;
AD1CON2bits.VCFG = 1;
and has the following Reference voltages:
AVDD = 3.2 Volts.
AVSS = 0 Volts.
VREF+ = 2.8 Volts.
VREF- = 0 Volts.
What voltage value (to three decimal places) does the following binary sequence represent?
1111 1010 1000 0000 0000 0000 0000 0000
2.74 0b1111101010/1024*2.8

HW12-3(PIC32-ADCVoltage-Fractional-32bit)
AD1CON1bits.FORM = 6;
AD1CON2bits.VCFG = 3;
and has the following Reference voltages:
AVDD = 2.3 Volts.
AVSS = 0 Volts.
VREF+ = 3.6 Volts.
VREF- = 0 Volts.
What voltage value (to three decimal places) does the following binary sequence represent?
0100 0010 1000 0000 0000 0000 0000 0000
0.935 0b100001010/1024*3.6=0.935

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 6;
AD1CON2bits.VCFG = 0;
and has the following Reference voltages:
AVDD = 2.6 Volts.
AVSS = 0 Volts.
VREF+ = 2.5 Volts.
VREF- = 0 Volts.
What voltage value (to three decimal places) does the following binary sequence represent?
1010 1000 0000 0000 0000 0000 0000 0000
Answer = 1.706

HW12-3(PIC32-ADCVoltage-Fractional-Difference-16bit)
Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 2;
AD1CON2bits.VCFG = 0;
and has the following Reference voltages:
AVDD = 4.6 Volts.
AVSS = 0 Volts.
VREF+ = 4.8 Volts.
VREF- = 0.3 Volts.
What voltage value (to three decimal places) does the following binary sequence represent?
0111 0000 1000 0000
2.021

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 2;
AD1CON2bits.VCFG = 3;
and has the following Reference voltages:
AVDD = 4 Volts.
AVSS = 0.9 Volts.
VREF+ = 4.6 Volts.
VREF- = 0.3 Volts.
What voltage value (to three decimal places) does the following binary sequence represent?
0101 0100 1100 0000
1.724 (0b101010011)/1024*(4.6-0.3)+0.3

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 2;
AD1CON2bits.VCFG = 3;
and has the following Reference voltages:
AVDD = 4 Volts.
AVSS = 1.3 Volts.
VREF+ = 3.9 Volts.
VREF- = 1.4 Volts.
What voltage value (to three decimal places) does the following binary sequence represent?
0001 0101 1000 0000
1.610

Note: Most of 12-3 makes use of this chart.

DIFFERENCE EQUATION SUMMARY	
Signed	Unsigned
((B±512)/(2^10))(V_{REFH}-V_{REFL})+(V_{REFL})	(B/(2^10))(V_{REFH}-V_{REFL})+(V_{REFL})

B = All *d* bits converted to binary. These are the bits referenced in the *FORM* command.
*Leading 1: -512 (signed)
*Leading 0: +512 (signed)
****With signed, include the sign bit in B**

	VREFH	VREFL
000	AVDD	AVSS
001	External VREF+ pin	AVSS
010	AVDD	External VREF- pin
011	External VREF+ pin	External VREF- pin
1xx	AVDD	AVSS

bit 10-8 FORM<2:0>: Data Output Format bits
111 = Signed Fractional 32-bit (DOUT = sddd dddd dd00 0000 0000 0000 0000 0000)
110 = Fractional 32-bit (DOUT = dddd dddd dd00 0000 0000 0000 0000 0000)
101 = Signed Integer 32-bit (DOUT = ssss ssss ssss ssss ssss sssd dddd dddd)
100 = Integer 32-bit (DOUT = 0000 0000 0000 0000 0000 00dd dddd dddd)
011 = Signed Fractional 16-bit (DOUT = 0000 0000 0000 0000 sddd dddd dd00 0000)
010 = Fractional 16-bit (DOUT = 0000 0000 0000 0000 dddd dddd dd00 0000)
001 = Signed Integer 16-bit (DOUT = 0000 0000 0000 0000 ssss sssd dddd dddd)
000 = Integer 16-bit (DOUT = 0000 0000 0000 0000 0000 00dd dddd dddd)

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 2;
AD1CON2bits.VCFG = 1;
and has the following Reference voltages:
AVDD = 4.4 Volts.
AVSS = 0.7 Volts.
VREF+ = 4.4 Volts.
VREF- = 0.8 Volts.
What voltage value (to three decimal places) does the following binary sequence represent?
1111 0110 0100 0000
4.259 0b1111011001/1024*(4.4-.7)+.7

HW12-3(PIC32-ADCVoltage-Fractional-Difference-32bit)
Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 1;
AD1CON2bits.VCFG = 3;
and has the following Reference voltages:
AVDD = 4.8 Volts.
AVSS = 0 Volts.
VREF+ = 3.1 Volts.
VREF- = 0 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
1111 1110 0001 0010
0.0545 0b10010*3.1/1024

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 6;
AD1CON2bits.VCFG = 2;
and has the following Reference voltages:
AVDD = 4.8 Volts.
AVSS = 0.1 Volts.
VREF+ = 3.6 Volts.
VREF- = 0.8 Volts.
What voltage value (to three decimal places) does the following binary sequence represent?
1100 1101 0100 0000 0000 0000 0000 0000
4.007
(0b1100110101 / 1024) * (4.8 - 0.8) - (-0.8)

HW12-3(PIC32-ADCVoltage-Signed-16bit)
Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 5;
AD1CON2bits.VCFG = 2;
and has the following Reference voltages:
AVDD = 3.5 Volts.
AVSS = 0 Volts.
VREF+ = 3.1 Volts.
VREF- = 0 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
1111 1111 1111 1111 1111 1111 1111 1011
1.7329

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 1;
AD1CON2bits.VCFG = 1;
and has the following Reference voltages:
AVDD = 2.1 Volts.
AVSS = 0 Volts.
VREF+ = 2.2 Volts.
VREF- = 0 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
1111 1111 0010 1011
0.6424
((0b1100101011 - 512) / 1024) * 2.2

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 1;
AD1CON2bits.VCFG = 3;
and has the following Reference voltages:
AVDD = 3.3 Volts.
AVSS = 0 Volts.
VREF+ = 2.6 Volts.
VREF- = 0 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
1111 1110 1110 0101
0.5814

HW12-3(PIC32-ADCVoltage-SignedFractional-16bit)
Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 3;
AD1CON2bits.VCFG = 1;
and has the following Reference voltages:
AVDD = 4.8 Volts.
AVSS = 0 Volts.
VREF+ = 4.8 Volts.
VREF- = 0 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
1001 0001 1100 0000
0.3328
((0b1001000111 - 512) / 1024) * 4.8

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 3;
AD1CON2bits.VCFG = 1;
and has the following Reference voltages:
AVDD = 3.8 Volts.
AVSS = 0 Volts.
VREF+ = 4.7 Volts.
VREF- = 0 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
1111 1011 1000 0000
2.2674

HW12-3(PIC32-ADCVoltage-SignedFractional-32bit)
Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 7;
AD1CON2bits.VCFG = 2;
and has the following Reference voltages:
AVDD = 5 Volts.
AVSS = 0 Volts.
VREF+ = 4.1 Volts.
VREF- = 0 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
1011 1111 0000 0000 0000 0000 0000 0000
1.2305
((0b1011111100 - 512) / 1024) * 5

HW12-3(PIC32-ADCVoltage-Signed-32bit)
Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 5;
AD1CON2bits.VCFG = 3;
and has the following Reference voltages:
AVDD = 4.6 Volts.
AVSS = 0 Volts.
VREF+ = 2.8 Volts.
VREF- = 0 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
1111 1111 1111 1111 1111 1110 1001 0101
0.4074
((0b1010010101 - 512) / 1024) * 2.8

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 5;
AD1CON2bits.VCFG = 2;
and has the following Reference voltages:
AVDD = 2.9 Volts.
AVSS = 0 Volts.
VREF+ = 3.7 Volts.
VREF- = 0 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
1111 1111 1111 1111 1111 1110 1101 1001
0.6146

HW12-3(PIC32-ADCVoltage-SignedDifference-16bit)
Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 1;
AD1CON2bits.VCFG = 1;
and has the following Reference voltages:
AVDD = 4.1 Volts.
AVSS = 1.2 Volts.
VREF+ = 3.7 Volts.
VREF- = 1.6 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
1111 1111 1101 1110
2.367
[((0b1111011110 - 512) / 1024) * (3.7 - 1.2)] + 1.2

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 1;
AD1CON2bits.VCFG = 0;
and has the following Reference voltages:
AVDD = 4.6 Volts.
AVSS = 1.2 Volts.
VREF+ = 2.8 Volts.
VREF- = 1.4 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
1111 1111 1111 0110
2.8668

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 1;
AD1CON2bits.VCFG = 3;
and has the following Reference voltages:
AVDD = 4.3 Volts.
AVSS = 0 Volts.
VREF+ = 4.4 Volts.
VREF- = 0.3 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
1111 1110 0011 0000
0.4922

HW12-3(PIC32-ADCVoltage-SignedDifference-32bit)
Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 5;
AD1CON2bits.VCFG = 3;
and has the following Reference voltages:
AVDD = 4.6 Volts.
AVSS = 1.5 Volts.
VREF+ = 5 Volts.
VREF- = 1.4 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
1111 1111 1111 1111 1111 1111 0100 1010
2.5602
[((0b1101001010 - 512) / 1024) * (5 - 1.4)] + 1.4

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 5;
AD1CON2bits.VCFG = 2;
and has the following Reference voltages:
AVDD = 3.7 Volts.
AVSS = 0.5 Volts.
VREF+ = 3.6 Volts.
VREF- = 0.7 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
1111 1111 1111 1111 1111 1111 0001 0000
1.4969

HW12-3(PIC32-ADCVoltage-SignedFractional-Difference-16bit)
Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 3;
AD1CON2bits.VCFG = 2;
and has the following Reference voltages:
AVDD = 3.8 Volts.
AVSS = 0.5 Volts.
VREF+ = 4.6 Volts.
VREF- = 1 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
1111 1010 1100 0000
2.3426 0b111101011/1024*(3.8-1)+1

HW12-3(PIC32-ADCVoltage-SignedFractionalDifference-32bit)
Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 7;
AD1CON2bits.VCFG = 2;
and has the following Reference voltages:
AVDD = 4.5 Volts.
AVSS = 1 Volts.
VREF+ = 4.6 Volts.
VREF- = 1.4 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
1110 1111 1000 0000 0000 0000 0000 0000
2.7502
((0b110111110 / 1024) * (4.5 - 1.4)) + 1.4

HW12-3(PIC32-ADCVoltage-SignedFractionalPositiveDifference-16bit)
(Binary value + 512)/1024*(V_{REFH}-V_{REFL})+V_{REFL}

HW12-3(PIC32-ADCVoltage-SignedFractionalPositiveDifference-32bit)
(Binary value + 512)/1024*(V_{REFH}-V_{REFL})+V_{REFL}

HW12-3(PIC32-ADCVoltage-SignedFractionalPositive-16bit)
Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 3;
AD1CON2bits.VCFG = 2;
and has the following Reference voltages:
AVDD = 3.2 Volts.
AVSS = 0 Volts.
VREF+ = 3.2 Volts.
VREF- = 0 Volts.
What voltage value does the following binary sequence represent?
0110 1001 1000 0000
2.9188
(Binary value + 512)/1024*(V_{REFH})

HW12-3(PIC32-ADCVoltage-SignedPositive-16bit)
Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 1;
AD1CON2bits.VCFG = 0;
and has the following Reference voltages:
AVDD = 3 Volts.
AVSS = 0 Volts.
VREF+ = 3.1 Volts.
VREF- = 0 Volts.
What voltage value does the following binary sequence represent (to four decimal places)?
0000 0001 1010 1011
2.751

HW12-3(PIC32-ADCVoltage-Unsigned-16bit)
bit 10-8 FORM<2:0>: Data Output Format bits
111 = Signed Fractional 32-bit (DOUT = sddd dddd dd00 0000 0000 0000 0000 0000)
110 = Fractional 32-bit (DOUT = dddd dddd dd00 0000 0000 0000 0000 0000)
101 = Signed Integer 32-bit (DOUT = ssss ssss ssss ssss ssss sssd dddd dddd)
100 = Integer 32-bit (DOUT = 0000 0000 0000 0000 0000 00dd dddd dddd)
011 = Signed Fractional 16-bit (DOUT = 0000 0000 0000 0000 sddd dddd dd00 0000)
010 = Fractional 16-bit (DOUT = 0000 0000 0000 0000 dddd dddd dd00 0000)
001 = Signed Integer 16-bit (DOUT = 0000 0000 0000 0000 ssss sssd dddd dddd)
000 = Integer 16-bit (DOUT = 0000 0000 0000 0000 0000 00dd dddd dddd)

Voltage = (binary converted to decimal / 1024) * Positive Voltage Reference

Consider a PIC32MX with the following ADC1 settings,

	VREFH	VREFL
000	AVDD	AVSS
001	External VREF+ pin	AVSS
010	AVDD	External VREF- pin
011	External VREF+ pin	External VREF- pin
1xx	AVDD	AVSS

AD1CON1bits.FORM = 0;
AD1CON2bits.VCFG = 1;
and has the following Reference voltages:
AVDD = 3.8 Volts.
AVSS = 0 Volts.
VREF+ = 3.5 Volts.
VREF- = 0 Volts.
What voltage value (to three decimal places) does the following binary sequence represent?
0000 0011 0111 1110
3.056
Since FORM = 0, convert the rightmost 10 bits to decimal.
1101111110 = 894
Since VCFG = 1, the positive reference voltage is VREF+.
(894/2^10) * 3.5 = 3.056

HW12-3(PIC32-ADCVoltage-Unsigned-32bit)
Same as HW 12-3(PIC32-ADCVoltage-Unsigned-16bit) above

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 4;
AD1CON2bits.VCFG = 0;
and has the following Reference voltages:
AVDD = 4.6 Volts.
AVSS = 0 Volts.
VREF+ = 3.9 Volts.
VREF- = 0 Volts.
What voltage value (to three decimal places) does the following binary sequence represent?
0000 0000 0000 0000 0000 0000 0010 1001
0.184

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 4;
AD1CON2bits.VCFG = 3;
and has the following Reference voltages:
AVDD = 2.1 Volts.
AVSS = 0 Volts.
VREF+ = 2.5 Volts.
VREF- = 0 Volts.
What voltage value (to three decimal places) does the following binary sequence represent?
0000 0000 0000 0000 0000 0001 1011 0111 439/2^10*2.5
1.072

HW12-3(PIC32-ADCVoltage-UnsignedDifference-16bit)
Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 0;
AD1CON2bits.VCFG = 3;
and has the following Reference voltages:
AVDD = 4.3 Volts.
AVSS = 2.5 Volts.
VREF+ = 4.9 Volts.
VREF- = 2.6 Volts.
What voltage value (to three decimal places) does the following binary sequence represent?
0000 00nn11 0000 1001
4.345

Consider a PIC32MX with the following ADC1 settings,
AD1CON1bits.FORM = 0;
AD1CON2bits.VCFG = 1;
and has the following Reference voltages:
AVDD = 4.5 Volts.
AVSS = 2.6 Volts.
VREF+ = 4.3 Volts.
VREF- = 1.5 Volts.
What voltage value (to three decimal places) does the following binary sequence represent?
0000 0000 0100 0110
2.716

HW12-3(PIC32-ADCVoltage-UnsignedDifference-32bit)
One more...

HW12-3(PIC32-ADCxCHS)
Give a single C instruction to configure a **PIC32MX's** ADC Module with the following properties:
- Channel 0 negative input is VREFL for MUX B.
- Channel 0 positive input is AN10 for MUX B.
- Channel 0 negative input is AN1 for MUX A.
- Channel 0 positive input is AN1 for MUX A.
AD1CHS = 0x0A810000;

HW12-3(PIC32-ADCxCON1)
Give a single C instruction for a PIC32MX which will tell the Analog to Digital peripheral module to output the sample as a fractional 32-bit value without modifying any other parameters. (Leave a single space between all variables and/or operators and use no parentheses.
AD1CON1bits.FORM = 6;

Use the info from the data sheet found under HW 12-3 (PIC32-ADCxCON1Multi)

Give a single C instruction which will turn off the Analog to Digital peripheral module in a PIC32MX without modifying any other parameters. (Leave a single space between all variables and/or operators and use no parentheses.)
AD1CON1bits.ON = 0;

HW12-3(PIC32-ADCxCON1Multi)
Give a single C instruction to configure a PIC32MX's ADC Module with the following properties:
-ADC module IS operating.
-Continue Module operation when the device enters Idle Mode.
-16-bit Signed Integer Output.
-Internal counter ends sampling and starts conversion.
-Sampling begins when SAMP bit is set.
-When SSRC = 000, writing a '0' to SAMP will end sampling and start conversion.
AD1CON1 = 0x81E0;

HW12-3(PIC32-ADCxCON2)
Give a single C instruction for a **PIC32MX** which will tell the Analog to Digital peripheral module to use **AVDD**and **AVSS** as its voltage references without modifying any other parameters. (Leave a single space between all variables and/or operators and use no parentheses.)
AD1CON2bits.VCFG = 0;

Use the info from the data sheet found under HW 12-3 (PIC32-ADCxCON2Multi)

Give a single C instruction for a PIC32MX which will tell the Analog to Digital peripheral module to interrupt after every five sample sequences without modifying any other parameters. (Leave a single space between all variables and/or operators and use no parentheses.)
AD1CON2bits.SMPI = 4;

HW12-3(PIC32-ADCxCON2Multi)
Give a single C instruction to configure a **PIC32MX's** ADC Module with the following properties:

- VREFH uses AVDD, VREFL uses external VREF- pin.
- Disable Offset Calibration Mode.
- Scan the Inputs.
- Interrupt after completion of each 3rd conversion.
- Buffer configured as one 16-word buffer.
- Always use Sample A MUX settings for input.

AD1CON2 = 0x4408;

Use the following

HW12-3(PIC32-ADCxCON3)
Give a single C instruction for a **PIC32MX** which will have the Analog to Digital peripheral module use a sample period of **four TAD**s without modifying any other parameters. (Leave a single space between all variables and/or operators and use no parentheses.)
AD1CON3bits.SAMC = 4;

Use the info from the data sheet found under HW 12-3(PIC32-ADCxCON3Multi)

HW12-3(PIC32-ADCxCON3Multi)

Give a single C instruction to configure a PIC32MX's ADC Module with the following properties:

- Clock derived from FRC.
- Sample time = 0 TAD.
- TAD = 2 TPB.
AD1CON3 = 0x8000;

Give a single C instruction to configure a **PIC32MX's** ADC Module with the following properties:

- Clock derived from FRC.
- Sample time = 5 TAD.
- TAD = 10 TPB.

AD1CON3 = 0x8504;

Give a single C instruction to configure a PIC32MX's ADC Module with the following properties:

- Clock derived from FRC.
- Sample time = 1 TAD.
- TAD = 6 TPB.
AD1CON3 = 0x8102

HW12-3(PIC32-ADCxCSSL)
Give a single C instruction to select the following pins as input for MUX A of a **PIC32MX's** ADC Module:

AN2
AN3
AN12
VSS

AD1CSSL = 0x900C;

If VSS is listed, bit 15 = 1.
If IVREF is listed, bit 14 = 1.
If CTMU is listed, bit 13 = 1.
Set each corresponding bit to 1 that's listed.

For the above problem, bit 15 = 1, bit 12 = 1, bit 3 = 1, and bit 2 = 1. The rest are 0.
1001 0000 0000 1100 = 0x900C

Give a single C instruction to select the following pins as input for MUX A of a PIC32MX's ADC Module:

AN8
AN11
AN12
CTMU

AD1CSSL = 0x3900;

HW12-4(PIC32-ADC-ProduceASequence)

Example 1
Complete the C instructions below to produce the following sampling sequence using ADC1 of a PIC32MX.
AD1BUF0 = MUXA AN3
AD1BUF1 = MUXB AN4
AD1BUF2 = MUXA AN3
AD1BUF3 = MUXB AN4
AD1BUF4 = MUXA AN3
AD1BUF5 = MUXB AN4
< Interrupt Generated >
AD1BUF8 = MUXA AN3
AD1BUF9 = MUXB AN4
AD1BUF10 = MUXA AN3
AD1BUF11 = MUXB AN4
AD1BUF12 = MUXA AN3
AD1BUF13 = MUXB AN4
< Interrupt Generated >
etc...
AD1CON2bits.CSCNA = **0**; ***1 if the ANx values change within any MUXy, 0 otherwise.
AD1CON2bits.SMPI = **5**; Amount of buffers between interrupts minus 1.
AD1CON2bits.BUFM = **1**;
Explanation 1: BUFM = 0 when conversion results are written sequentially starting at ADC1BUF0 and goes until the number of samples defined by SMPI and starts over after the interrupt
AD1CON2bits.ALTS = **1**; If they alternate between MUXA and MUXB, this is =1, otherwise 0.
AD1CHSbits.CH0SB = **4**; If the MUXB values correspond to one AN value (like this example), this is just equal to the value of that ANx. If it doesn't correspond to just one, this is =X.
AD1CHSbits.CH0SA = **3**; Same thing as CH0SB, but for MUXA commands.
AD1CSSL = **X**; ***Use the CSSL reference and put a 1 for every ANx value used. 0s when you don't. Unless it's only 1 value, then it's an X

Example 2
Complete the C instructions below to produce the following sampling sequence using ADC1 of a PIC32MX .
AD1BUF0 = MUXA AN0
AD1BUF1 = MUXA AN5
AD1BUF2 = MUXA AN6
AD1BUF3 = MUXA AN7
AD1BUF4 = MUXA AN0
< Interrupt Generated >
AD1BUF0 = MUXA AN5
AD1BUF1 = MUXA AN6
AD1BUF2 = MUXA AN7
AD1BUF3 = MUXA AN0
AD1BUF4 = MUXA AN5
< Interrupt Generated >
etc...
AD1CON2bits.CSCNA = **1**;
AD1CON2bits.SMPI = **4**;
AD1CON2bits.BUFM = **0**;
AD1CON2bits.ALTS = **0**;
AD1CHSbits.CH0SB = **X**;
AD1CHSbits.CH0SA = **X**;
AD1CSSL = **0xE1**;

Example 3
Complete the C instructions below to produce the following sampling sequence using ADC1 of a PIC32MX .
AD1BUF0 = MUXA AN10
AD1BUF1 = MUXB AN2
AD1BUF2 = MUXA AN10
AD1BUF3 = MUXB AN2
AD1BUF4 = MUXA AN10
AD1BUF5 = MUXB AN2
AD1BUF6 = MUXA AN10
< Interrupt Generated >
AD1BUF8 = MUXB AN2
AD1BUF9 = MUXA AN10
AD1BUF10 = MUXB AN2
AD1BUF11 = MUXA AN10
AD1BUF12 = MUXB AN2
AD1BUF13 = MUXA AN10
AD1BUF14 = MUXB AN2
< Interrupt Generated >
Etc…
AD1CON2bits.CSCNA = **0**;
AD1CON2bits.SMPI = **6**;
AD1CON2bits.BUFM = **1**;
AD1CON2bits.ALTS = **1**;
AD1CHSbits.CH0SB = **2**;
AD1CHSbits.CH0SA = **10**;
AD1CSSL = **X**;

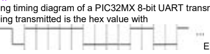
Example 4
Complete the C instructions below to produce the following sampling sequence using ADC1 of a PIC32MX .
AD1BUF0 = MUXA AN0
AD1BUF1 = MUXB IVREF
AD1BUF2 = MUXA AN2
< Interrupt Generated >
AD1BUF8 = MUXB IVREF
AD1BUF9 = MUXA AN4
AD1BUF10 = MUXB IVREF
< Interrupt Generated >
AD1BUF0 = MUXA AN5
AD1BUF1 = MUXB IVREF
AD1BUF2 = MUXA AN7
< Interrupt Generated >
AD1BUF8 = MUXB IVREF
AD1BUF9 = MUXA AN0
AD1BUF10 = MUXB IVREF
< Interrupt Generated >
etc...
AD1CON2bits.CSCNA = **1**;
AD1CON2bits.SMPI = **2**;
AD1CON2bits.BUFM = **1**;
AD1CON2bits.ALTS = **1**;
AD1CHSbits.CH0SB = **14**;
AD1CHSbits.CH0SA = **X**;
AD1CSSL = **0x00B5**;

HW12-4(PIC32-ADC-WhatIsSequence)
Given the following register settings in a PIC32MX, give the sampling sequence that takes place showing what buffer(s) is/are filled by what input (including which MUX) and show when an interrupt would be generated.
AD1CON2 = 0x00000007
AD1CHS = 0x83880000
AD1CSSL = 0x000000CD

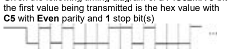
Chapter 13
HW13-2(RS-232-General)
What does Modem stand for? **Modulate-Demodulate**
What does CTS in RS-232 stand for? **Clear to Send**
What does DTR in RS-232 stand for? **Data Terminal Ready**
What does RTS in RS-232 stand for? **Request to Send**
What does DSR in RS-232 stand for? **Data Set Ready**
What was RS-232 and modems originally designed to communicate over? **Telephone Lines**
What does RI in RS-232 stant for? **Ring Indicator**
What does RxD in RS-232 stand for? **Receive Data**
What does DTE in RS-232 stand for? **Data Terminal Equipment**
What does DCD in RS-232 stand for? **Data Carrier Detect**
What's the minimum number of wires necessary to communicate with the **RS-232** protocol? **3**

HW13-3(UART-General)
What does Baud Rate mean? **Baud rate is the "symbols per second"**
How many Start Bits does a UART have? **One**
How many Stop Bits does a UART have? **Variable/programmable**
What is the purpose of a UART? **It transmits and receives serial data using a shift register**
What is a 9th UART bit usually used for? **The 9th bit determines whether or not the remaining**
8 bits transmitted contains a device address, or data for the selected device.
What does UART stand for? **Universal Asynchronous Receiver Transmitter**
What type of communication does UART perform? **Asynchronous Serial**
How many start bits does a UART have? **One**
What causes a UART Overrun Error? **If a new byte arrives before the byte in the buffer is moved into the CPU, an Overrun Error occurs.**

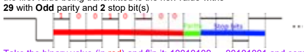
HW13-3(UART-WhatIsTransmission-Parity)
Explanation below
Given the following timing diagram of a PIC32MX 8-bit UART transmission with parity, the first value being transmitted is the hex value with

1F with **Odd** ( Even/Odd) parity and **2** stop bit(s)

Given the following timing diagram of a PIC32MX 8-bit UART transmission with parity, the first value being transmitted is the hex value with
C5 with **Even** parity and **1** stop bit(s)



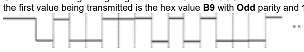
Given the following timing diagram of a PIC32MX 8-bit UART transmission with parity, the first value being transmitted is the hex value withs
29 with **Odd** parity and **2** stop bit(s)



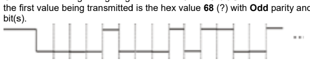
Take the binary value (in red) and flip it: 10010100 → 00101001 and convert that to a hex value. This the data being transmitted.
The parity bit is **ODD** if there is an odd number of bits that are 1 in the transmission plus the parity bit if it is 1, and it's **EVEN** if there is an even number of bits that are 1 in the transmission plus the parity bit if it is 1. For the above example, there are three 1's and you don't add the parity bit because it is 0, which gives us 3, an odd number, therefore it is an odd parity.
The number of stop bits is just the number of high values in the blue section before the next low, which is 2 in this case.
Given the following timing diagram of a PIC32MX 8-bit UART transmission with parity, the first value being transmitted is the hex value **0x34** with **Odd** (Even/Odd) parity and **1** stop bit(s).

Given the following timing diagram of a PIC32MX 8-bit UART transmission with parity, the first value being transmitted is the hex value **0xA1** with **Even** (Even/Odd) parity and **1** stop bit(s).

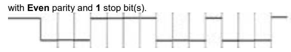
Given the following timing diagram of a **PIC32MX 8-bit UART** transmission ***with*** parity, the first value being transmitted is the hex value **B9** with **Odd** parity and **1** stop bit(s).



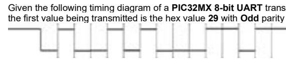
Given the following timing diagram of a **PIC32MX 8-bit UART** transmission ***with*** parity, the first value being transmitted is the hex value **68** (?) with **Odd** parity and **2** stop bit(s).



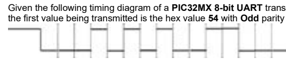
Given the following timing diagram of a **PIC32MX 8-bit UART** transmission ***with*** parity, the first value being transmitted is the hex value **3C**
with **Even** parity and **1** stop bit(s).



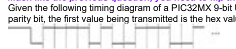
Given the following timing diagram of a **PIC32MX 8-bit UART** transmission ***with*** parity, the first value being transmitted is the hex value **29** with **Odd** parity and **2** stop bit(s).



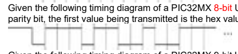
Given the following timing diagram of a **PIC32MX 8-bit UART** transmission ***with*** parity, the first value being transmitted is the hex value **54** with **Odd** parity and **2** stop bit(s).



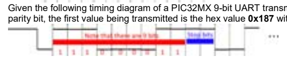
HW13-3(UART-WhatIsTransmission)
Much like the previous question, you need to flip the binary value (shown in red).
Given the following timing diagram of a PIC32MX 9-bit UART transmission with no parity bit, the first value being transmitted is the hex value **0x2F** with **2** stop bits.



Given the following timing diagram of a PIC32MX 8-bit UART transmission with no parity bit, the first value being transmitted is the hex value **0x37** with **1** stop bits.



Given the following timing diagram of a PIC32MX 9-bit UART transmission with no parity bit, the first value being transmitted is the hex value **0x187** with **2** stop bits.



HW13-4(PIC32-UART-UxMODE)
Give a single C instruction to configure UART 1 of a PIC32MX with the following parameters:(Assume all parameters not mentioned are 0's.)
Discontinue UART operation when the device enters Idle Mode. (bit13 = 1)
UxTX, UxRX, and UxBCLK pins are enabled. (bits9-8 = 11)
Loopback Mode Enabled. (bit6 = 1)
8-bit Data, No Parity. (bits2-1 = 00)
2 Stop Bits. (bit0 = 1)

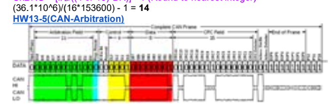
U1MODE = 0x2341;

HW13-4(PIC32-UART-UxSTA)
Give a single C instruction to configure UART 2of a PIC32MX with the following parameters:(Assume all parameters not mentioned are 0's.)
Automatic Address Detection is ENabled. (bit24 = 1)
Automatic Address Are: (bits23-16 = 1000 0001)
bit 0 = 1
bit 1 = 0
bit 2 = 0
bit 3 = 0
bit 4 = 0
bit 5 = 0
bit 6 = 0
bit 7 = 1
Interrupt is generated when transmit buffer is empty. (bits15-14 = 10)
Send Break on next Transmission. (bit11 = 1)
Address Detect Mode is Enabled. (bit5 = 1)

U2STA = 0x1818820;

HW13-4(PIC32-UART-WhatIsBaudRate)
1 => 4, 0 => 16
What is the baud rate (in bits/s to two decimal places) of a PIC32MX's UART 2 given the following C statements if fPBCLK is 45.90 MHz?
U2MODEbits.BRGH = 1; (Rate = 4) **There are only 2 modes, rate=4 or rate=16
U2BRG = 27653;
BR = f_{PBCLK} / [(4 or 16)*(UxBRG + 1)]
(45.9*10^6)/(4*(27653 + 1)) = **414.95** bits/s

HW13-4(PIC32-UART-WhatIsUxBRG)
What does a PIC32MX's UxBRG register need to be to give a baud rate of 153600 bps assuming a fPBCLK of 36.10 MHz and BRGH = 0? (Rate = 16)
UxBRG = [f_{PB}/((4 or 16)*BR)] - 1 (Round to nearest integer)
(36.1*10^6)/(16*153600) - 1 = **14**
HW13-5(CAN-Arbitration)



The bits below show the data of a CAN transmission for various nodes wanting to communicate on the bus all at the same time. Which node will get to use the bus first?
Node A: 001001000000000000001000000.... (576)
Node B: 001001010001000000011100000.... (593)
Node C: 001001000111000001011110000.... (583)
Node D: 001001111101000010011100000.... (637)
Node E: 001001100000000100010010000.... (608)
Node **A** will get control of the bus. (Lowest Value = Highest Priority)

HW13-5(CAN-Format-ACK-Delimeter)
Given the data of the CAN transmission shown..
0100101011010000001111010010011111010111010111111111110
...the value of the ACK Delimeter is **1** (answer in gray)

HW13-5(CAN-Format-ACK)
Given the data of the CAN transmission shown..
0111000101100000001001011011000100011101000001111111110
...the value of the ACK field is **0**

HW13-5(CAN-Format-CRC-Delimeter)
Given the data of the CAN transmission shown..
0101011100000000001001111000111000110101100101111111110
...the value of the CRC Delimeter is **0**

HW13-5(CAN-Format-CRC)
Given the data of the CAN transmission shown..
0111001110010000001110011010110000111101100011111111110
...the value of the CRC bits is **0x30F6** (in hex).

HW13-5(CAN-Format-Data)
Given the data of the CAN transmission shown..
0010000001110000001010001000001101110110101000111111110
...the value of the Data bits is **0x44** (in hex).

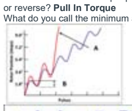
HW13-5(CAN-Format-DataLength)
Given the data of the CAN transmission shown
0011110010100001000111001111110001110001100101111111110
...the value of the Data Length is **8** (if binary value is greater than 8, the answer is 8)

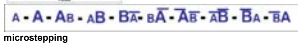
HW13-5(CAN-Format-ID)
Given the data of the CAN transmission shown..
0011101110010000001011100100010011100111011001111111110
...the value of the Identifier is **0x3B9** (in hex).

HW13-5(CAN-Format-RR)
Given the data of the CAN transmission shown..
0111000001111000001101000010110001101111101001111111110
...the value of the Request Remote field is **1**

Test 3 only covers parts 10 - 13

Chapter 19
HW19(StepperMotors-General)
What do you call the part of a stepper motor that is generally stationary and is held in place by the outer case? **Stator**
What type of stepper motor generally has the smallest step angles? **Hybrid Stepper Motor**
What type of stepper motor has a commuator to produce a magnetic field on the rotor? **None**
What do you call the part of a stepper motor that are radial projections on the mtor's poles used to concentrate the magnetic flux to certain positions on the stator? **Stator Teeth**
What is the following stepper motor graph exhibiting by label 'A'? **Loss of Synchronization**
What is the maximum torque produced by a stepper motor where it can still stop, start, or reverse? **Pull In Torque**
What do you call the minimum angle a stepper motor can rotate? **Step size**





microstepping

HW19(StepperMotors-Hybrid-RotorTeeth)
From the notes: "The number of Rotor Teeth is somewhat larger than the Stator Teeth."
So take the stator teeth number and go to the next highest amount.
Given a hybrid stepper motor with 4 poles and 40 stator teeth, which number would be the best
choice for the number of rotor teeth the motor has? **50**
Given a hybrid stepper motor with **3** poles and **36** stator teeth, which number would be the best choice for the number of rotor teeth the motor has? **46**
Given a hybrid stepper motor with 2 poles and 40 stator teeth, which number would be the best choice for the number of rotor teeth the motor has? **40**
Given a hybrid stepper motor with 5 poles and 40 stator teeth, which number would be the best choice for the number of rotor teeth the motor has? **50**
Given a hybrid stepper motor with 3 poles and 24 stator teeth, which number would be the best choice for the number of rotor teeth the motor has? **48**
Given a hybrid stepper motor with 8 poles and 24 stator teeth, which number would be the best choice for the number of rotor teeth the motor has? **32**
Given a hybrid stepper motor with 3 poles and 60 stator teeth, which number would be the best choice for the number of rotor teeth the motor has? **70**
HW19(StepperMotors-Hybrid-StatorTeeth)
From note: "The Stator Teeth are a multiple of the poles and typically are 16 to 32."
Pick closest multiple of # of poles that is less than # of rotor teeth
Given a hybrid stepper motor with 4 poles and 28 rotor teeth, which number would be the best choice for the number of stator teeth the motor has? **20**
Given a hybrid stepper motor with 5 poles and 20 rotor teeth, which number would be the best choice for the number of stator teeth the motor has? **15**
Given a hybrid stepper motor with 2 poles and 30 rotor teeth, which number would be the best choice for the number of stator teeth the motor has? **21**
Given a hybrid stepper motor with 6 poles and 20 rotor teeth, which number would be the best choice for the number of stator teeth the motor has? **12**
Given a hybrid stepper motor with 4 poles and 44 rotor teeth, which number would be the best choice for the number of stator teeth the motor has? **36**
Given a hybrid stepper motor with 5 poles and 50 rotor teeth, which number would be the best choice for the number of stator teeth the motor has? **35**

HW19(StepperMotors-Hybrid-StepSize)
Step size = 360*(RT-ST)/(RT*ST)
Given a 4-phase Hybrid stepper motor with 72 stator teeth and 90 rotor teeth, what is the step size of the motor (in degrees to two decimal places)? **1.00**

HW19(StepperMotors-Hybrid-ToothPitch)
Tooth pitch = 360/RT
Given a 5-phase Hybrid stepper motor with 39 stator teeth and 49 rotor teeth, what is the tooth pitch of the motor (in degrees to two decimal places)? **7.35**
Given a 3-phase Hybrid stepper motor with 21 stator teeth and 35 rotor teeth, what is the tooth pitch of the motor (in degrees to two decimal places)? **10.29**

HW19(StepperMotors-VR-MS-PitchStepSize)
Step size = 360/(phase*RT)
Given a 5-phase Multi-stack Variable Reluctance stepper motor with 20 rotor teeth, what is the step size of the motor (in degrees to two decimal places)?**3.6**

HW19(StepperMotors-VR-MS-StatorTeeth)
Number of Stator Teeth equals Number of Rotor Teeth because the offset of each stack from the others are what produce the next step angle.

Given a multi-stack variable-reluctance stepper motor with 4 phases and 32 stator teeth, how many rotor teeth must the motor have? **32**

HW19(StepperMotors-VR-SS-PitchStepSize)
Tooth pitch = 360/RT
Given a 5-phase Single-stack Variable Reluctance stepper motor with 8 rotor teeth, what is the tooth pitch of the motor (in degrees to two decimal places)? **45**

HW19(StepperMotors-VR-SS-RotorTeeth)
Not the same number but the next closest (greater than or less than)
Given a single-stack variable-reluctance stepper motor with 6 phases and 18 stator teeth, which number would be the best choice for the number of rotor teeth the motor has? **12**

HW19(StepperMotors-VR-SS-StatorTeeth)
Next closest number that is a multiple of the number of phases (greater than or less than)
Given a single-stack variable-reluctance stepper motor with 2 phases and 8 rotor teeth, which number would be the best choice for the number of stator teeth the motor has? **8**

Given a single-stack variable-reluctance stepper motor with 5 phases and 20 rotor teeth, which number would be the best choice for the number of stator teeth the motor has? **30**