***GPIO:***  
P1->SEL0 &= ~0x12; (0b00010010)

***SysTik:***  
- 24 Bit counter, counts down, 3MHz, decreases every 333.3us. counts from n->0, flag is set when count 1-0

*Initlize:*

SysTick -> CTRL = 0; disable SysTick During step

SysTick -> LOAD = 0x00FFFFFF;max reload value

SysTick -> VAL = 0; any write to current clears it

SysTick -> CTRL = 0x00000005; enable systic, 3MHz, No Interrupts (change to 7 for interrupts)

*Delay:*

SysTick -> LOAD = ((delay \* 3000) - 1); 1msec delay

SysTick -> VAL = 0; any write to CVR clears it

while((SysTick - > CTRL & 0x00010000) == 0);wait for flag to be SET

EXAMPLE: What to load systick if we want 1ms delay?

P1->SEL1 &= ~0x12;

P1->DIR &= ~0x12; make p1.4 & p1.1 inputs

P1->REN |= 0x12; enable pull resistors

P1->OUT |= 0x12; set as pullup

P1->IES |= 0x12; set pin interrupt to trigger when high to low

P1->IE |= 0x12; enable interrupt for pins

P1->IFG = 0; clear interrupt flags

***ADC Example:***

Graphical user interface, text, application, chat or text message

Description automatically generated  
Text

Description automatically generated

*Configure GPIO for PWM output*

*P2->DIR |= BIT4;* // P2.4 set TA0.1

*P2->SEL0 |= BIT4;*

*P2->SEL1 &= ~(BIT4);*

*TIMER\_A0->CCR[0] = 1000 - 1;* // PWM Period (# cycles of clock)

*TIMER\_A0->CCTL[1] = 0b0000 0000 0111 000*

// CCR1 reset/set mode 7

*TIMER\_A0->CCR[1] = 500;* // CCR1 PWM duty cycle in 10ths of percent

*TIMER\_A0->CTL = 0b0000 0010 0001 0100*; // SMCLK, Up Mode, clear TAR to start

*TIMER\_A0->CTL = TASSEL\_2 | MC\_1 | TACLR* // SMCLK, Up Mode, clear TAR to start

*TIMER\_A0->CTL = 0x0214;* // SMCLK, Up Mode, clear TAR to start



***Timer\_A, PWM, & ADC:***

4 clock sources, 2 stages of clk divider, counts up to 2^16 or rollover, can generate pwm

A screenshot of a computer

Description automatically generated with medium confidence

Shape

Description automatically generated

Shape

Description automatically generated

Diagram

Description automatically generated

Graphical user interface, diagram

Description automatically generated

Duty Cycle = Ton/Period

***MSP432 General Facts:***

5 clocks from 7 sources (2 external, 5 internal)

10kHz to 48 MHz operating frequency range

A screenshot of a computer

Description automatically generated with medium confidence

The maximum period of TAxCCRn when it counts by 1 every 0.333μs (for a 3MHz clock) is ~21.8 ms. Anything higher than 45.9Hz will result in errors.

Max voltage for a pin is 3.7 as input pin

Max data bits in one bus cycle is 32

Max current for most pins is 6mA

Address bus is 32 bits

Debounce -> Low pass filter

***Timer\_A Capture Mode Configuration***

***Quizzes & Answers:***

Power is an input to the Microcontroller on your MSP432 development board. Name two other kinds of inputs: Common ground (Pullup resistors)/ digital inputs & Analog sensors

The MSP432 uses the Harvard architecture which has two separate main busses to get data into the core. What kind of data does each buss retrieve? Instructions (Register Data P1\_>DIR) Data (Program flow, the executable .o file produced when building the project)

IF TIMER\_A2 is used At 3 MHz, INPUT divider = /4, what is the time between counts?

333ns\*4 = 1.332 uS

Change resolution of ADC -> change the full scale voltage & number of bits

One ADC core on MSP432

eUSCI -> enhanced universal serial interface

***Architecture:***

Harvard Architecture

* Separate memory for data and instructions
* Two sets of addressex\data between cpu and memory

CISC: Complex Instruction Set Computer, Intel x86

RISC: Reduced Instruction Set Computer, ARM, PIC, AVR

RISC is favored with the reduced price of ram

MSP432 (RISC)

***Communication Protocalls:***  
*UART – Universal Async transmit receive*

Async to another device, timing is based on the baud rate of the eUSCI\_A

Start bit, 7 | 8 data bits, even/odd/no parity bit, address bit, one or two stop bits

A screen with numbers and letters on it

Description automatically generated with low confidence

N = f / baudRate

Text

Description automatically generated

Text

Description automatically generated



***I2C –Inter Integrated Circuit:***

serial data line (SDA) and a serial clock line (SCL), device address, master, slave, 8-bit orientated, 100kbit/s, 400kbits/s, 5Mbits/s

The master/slave can only drive a signal down to 0v for logic 0.

Table

Description automatically generated

If slave exists it will respond with ACK bit, active low.

MSB is first.

4 I2C on MSP432

***SPI – Serial Peripheral Interface:***

MOSI, MISO

SS (Slave Select Line). This is a pin that selects what device you would like to use.

Good for only using one sensor with it, faster than I2C, full duplex, I2C is not, Does not define a speed limit. SPI is better for data transfer streams. I2C is better at multi master/slave applications.