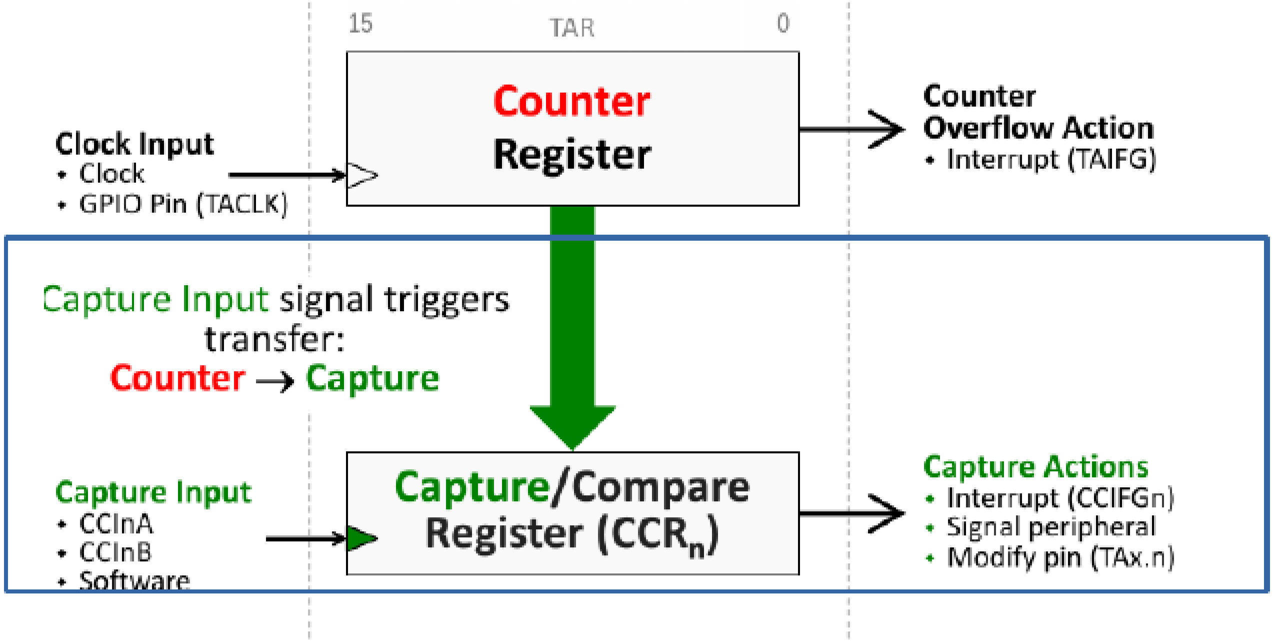
M6 Assignment

Timer0_A3 Capture Mode Example

Using CCR2 to hold captured value

Using the second register with the **Capture** feature allows the contents of the **Counter** to be captured (the **Counter** value is copied into the **Capture** register with no latency and very low power)

When a **Capture Input** signal occurs, the value from the Counter Register (**TAR**) is copied into the capture register (i.e. **CCR**)



15 TAR 0

Counter Register

Clock Input
- Clock
- GPIO Pin (TACLK)

Counter Overflow Action
- Interrupt (TAIFG)

Capture Input signal triggers transfer:
Counter → Capture

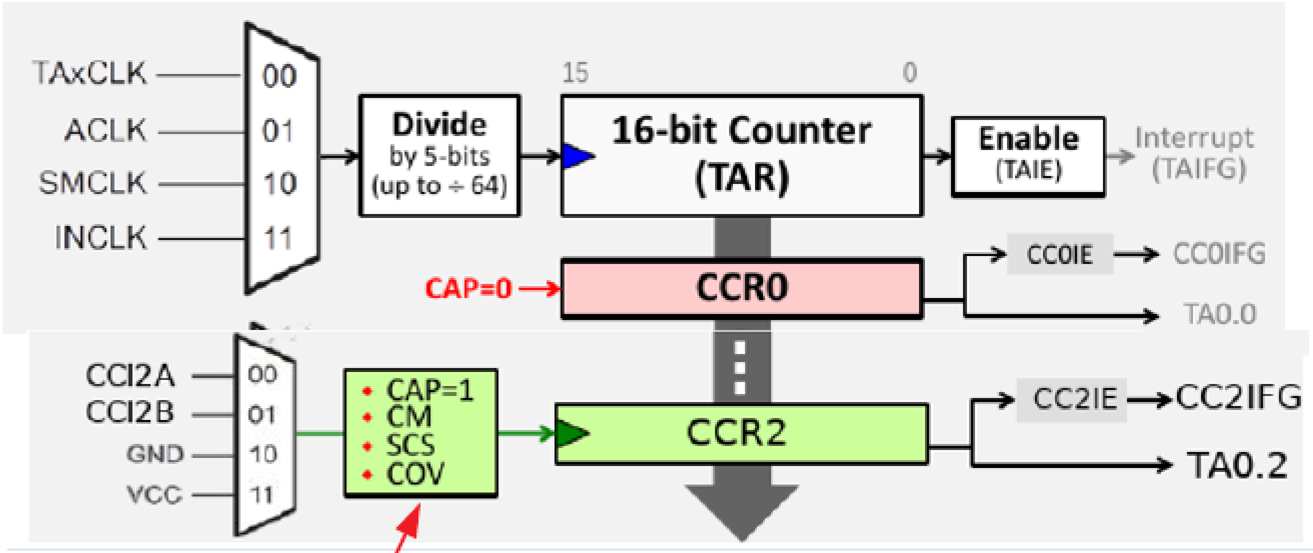
Capture/Compare Register (CCR_n)

Capture Input
- CCInA
- CCInB
- Software

Capture Actions
- Interrupt (CCIFGn)
- Signal peripheral
- Modify pin (TAx.n)

Notes
- Capture time (i.e. count value) when Capture Input signal occurs
- When capture is triggered, count value is placed in CCR and an interrupt is generated
- Capture Overflow (COV): indicates 2nd capture to CCR before 1st was read

Timer_A3 Capture Mode Summary

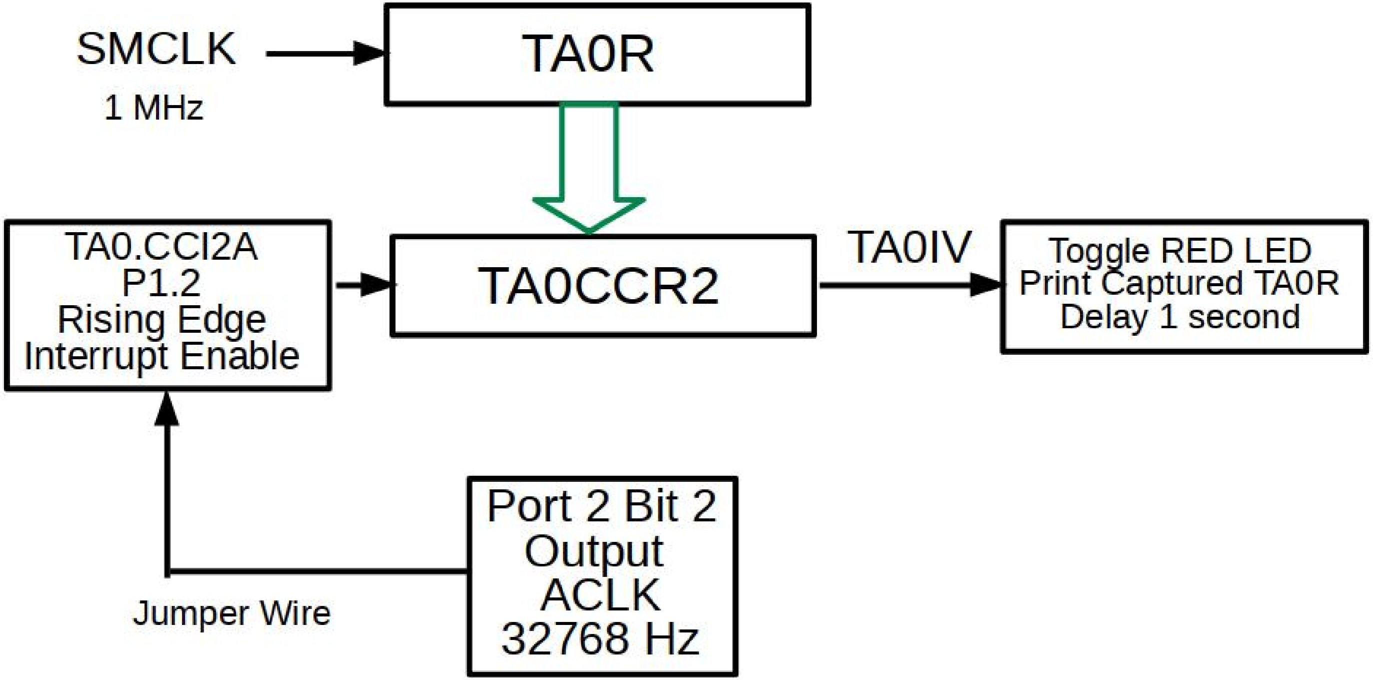


- **Capture or Compare (CAP)**
  CAP=1 for capture
- **Which Edge (CM)**
  Rising, Falling, or Both
- **Sync'd to Clock (SCS)**
  Is capture sync or async?
- **Capture Overflow (COV)**
  Did you miss a capture?

Measure 'split' times

i.e. Capture the value of the TAR when Input signal occurs

M6 Assignment
TA0 Capture Mode Example

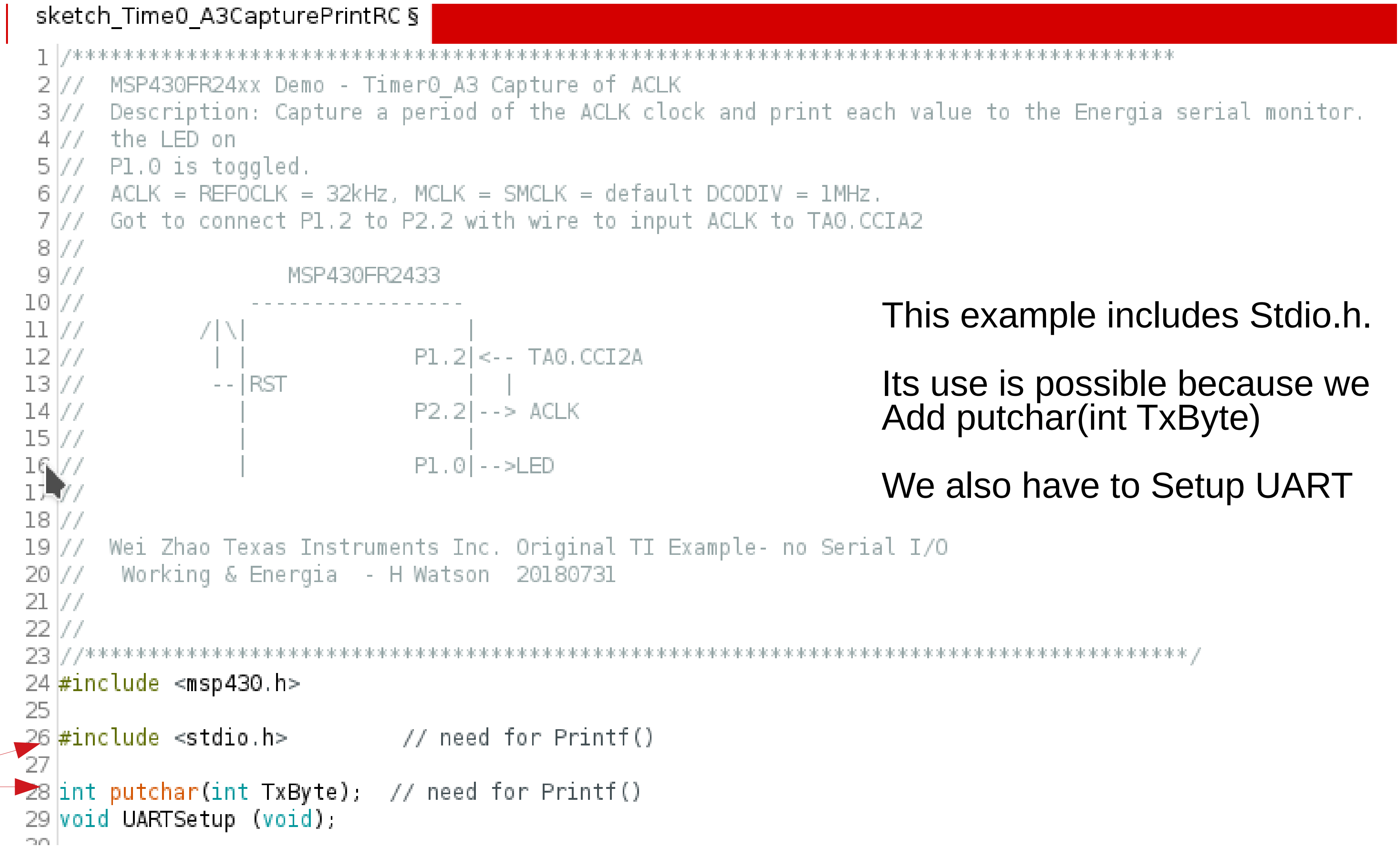


Capture Mode – Every second + ACLK duration

Capture TA0R count in microseconds

ACLK duration = 30.5 uSec   (1sec/32768)

(1sec = 1,000,000uSec / 2^16 [16 bits]) = 15.2587890625
The remainder = 1,000,000 – 983,040 = 16960



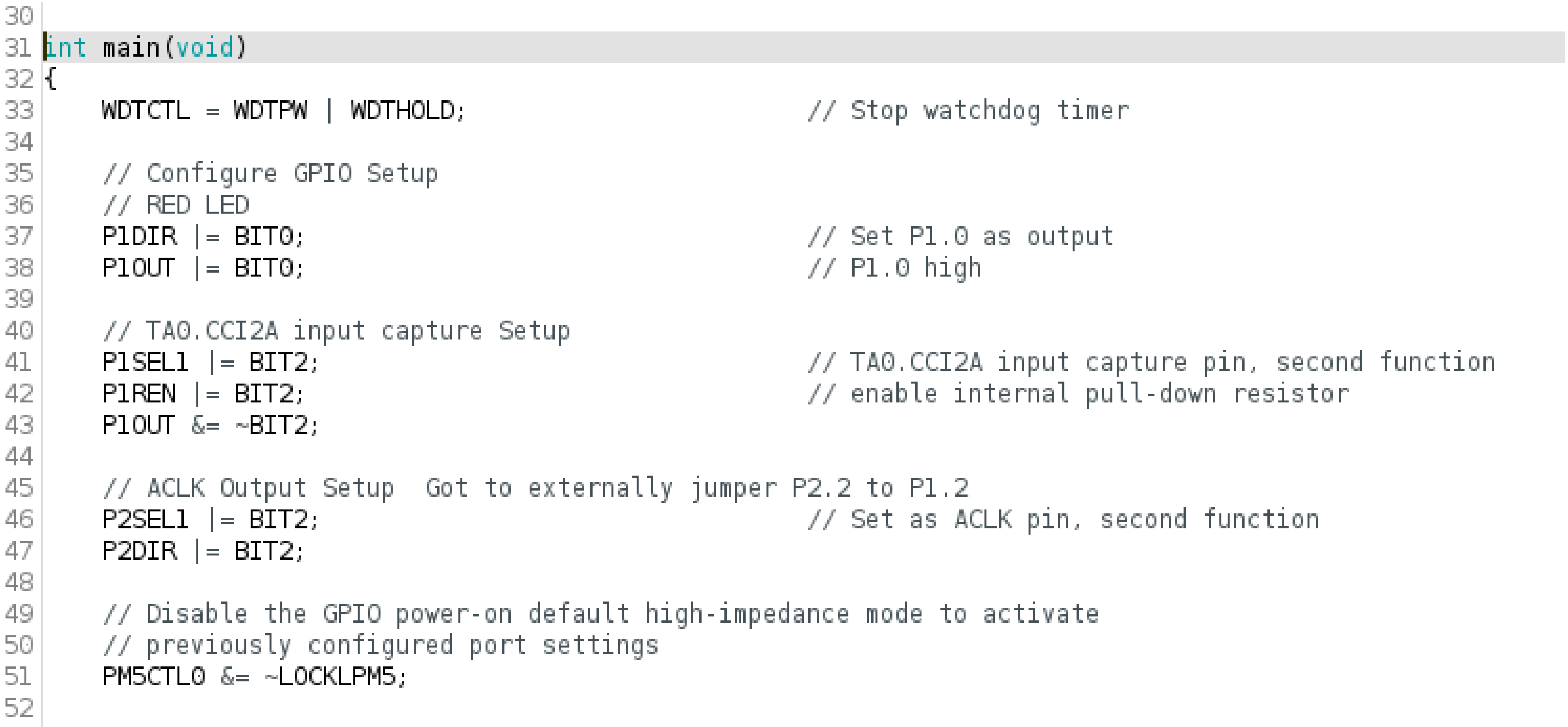
sketch_Time0_A3CapturePrintRC §

```
/***************************************************************************************
//  MSP430FR24xx Demo - Timer0_A3 Capture of ACLK
//  Description: Capture a period of the ACLK clock and print each value to the Energia serial monitor.
//  the LED on
//  P1.0 is toggled.
//  ACLK = REFOCLK = 32kHz, MCLK = SMCLK = default DCODIV = 1MHz.
//  Got to connect P1.2 to P2.2 with wire to input ACLK to TA0.CCIA2
//
//                MSP430FR2433
//             -----------------
//         /|\|                 |
//          | |             P1.2|<-- TA0.CCI2A
//          --|RST              |  |
//            |             P2.2|--> ACLK
//            |                 |
//            |             P1.0|-->LED
//
//
//  Wei Zhao Texas Instruments Inc. Original TI Example- no Serial I/O
//   Working & Energia  - H Watson  20180731
//
//
//***************************************************************************************/
#include <msp430.h>

#include <stdio.h>         // need for Printf()

int putchar(int TxByte);  // need for Printf()
void UARTSetup (void);
```

This example includes Stdio.h.

Its use is possible because we Add putchar(int TxByte)

We also have to Setup UART



```c
int main(void)
{
    WDTCTL = WDTPW | WDTHOLD;                       // Stop watchdog timer

    // Configure GPIO Setup
    // RED LED
    P1DIR |= BIT0;                                  // Set P1.0 as output
    P1OUT |= BIT0;                                  // P1.0 high

    // TA0.CCI2A input capture Setup
    P1SEL1 |= BIT2;                                 // TA0.CCI2A input capture pin, second function
    P1REN |= BIT2;                                  // enable internal pull-down resistor
    P1OUT &= ~BIT2;

    // ACLK Output Setup  Got to externally jumper P2.2 to P1.2
    P2SEL1 |= BIT2;                                 // Set as ACLK pin, second function
    P2DIR |= BIT2;

    // Disable the GPIO power-on default high-impedance mode to activate
    // previously configured port settings
    PM5CTL0 &= ~LOCKLPM5;
```

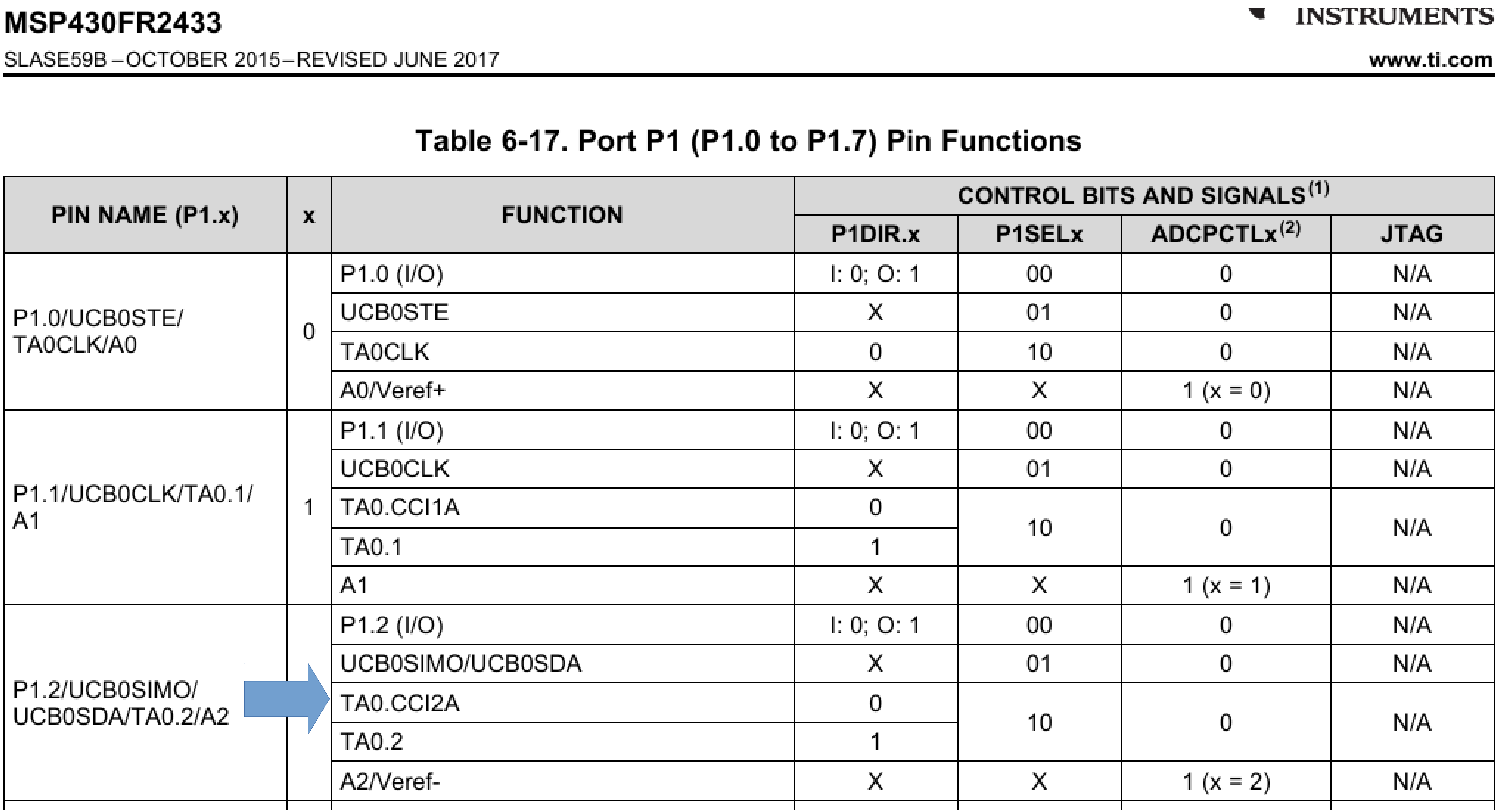
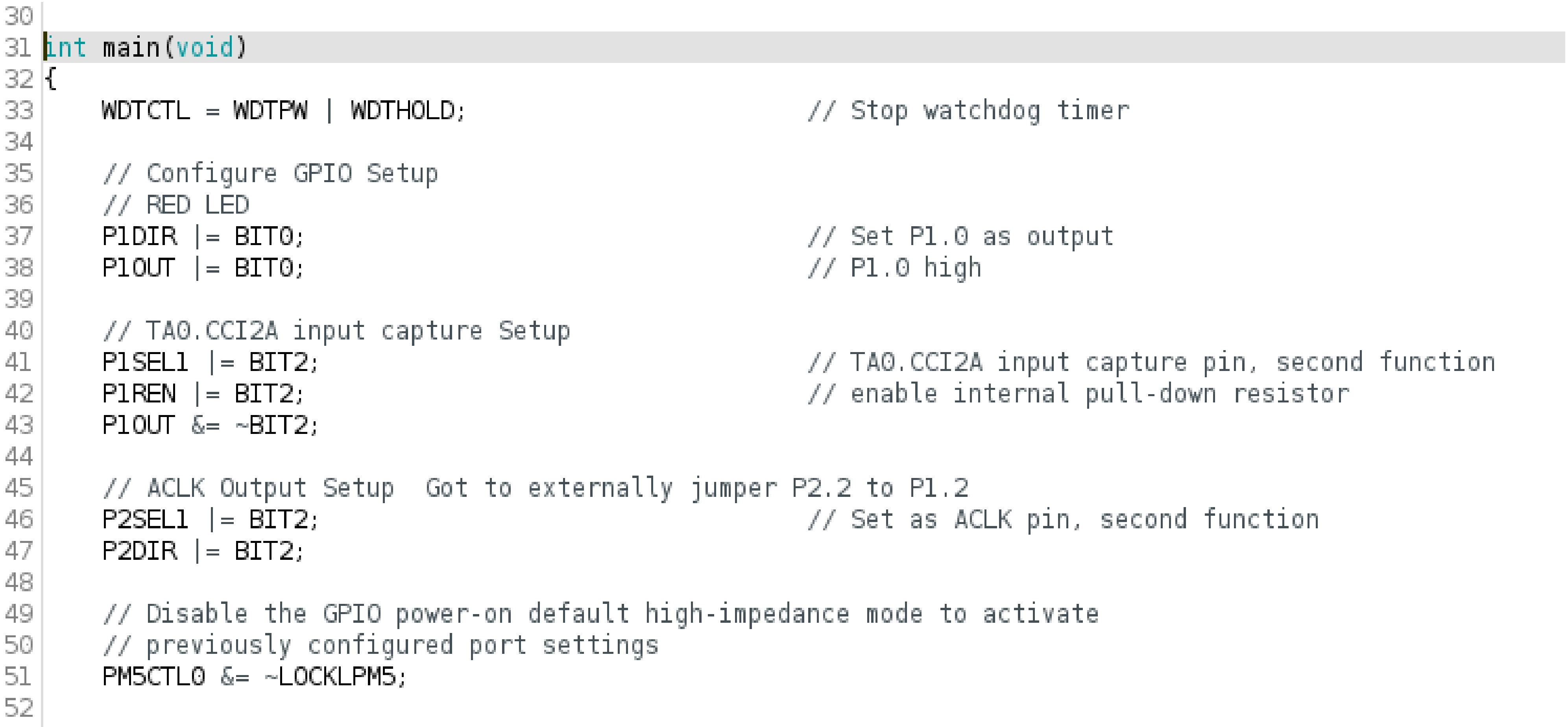


Table 6-17. Port P1 (P1.0 to P1.7) Pin Functions

PIN NAME (P1.x)	x	FUNCTION	CONTROL BITS AND SIGNALS(1)			
			P1DIR.x	P1SELx	ADCPCTLx(2)	JTAG
P1.0/UCB0STE/TA0CLK/A0	0	P1.0 (I/O)	I: 0; O: 1	00	0	N/A
		UCB0STE	X	01	0	N/A
		TA0CLK	0	10	0	N/A
		A0/Veref+	X	X	1 (x = 0)	N/A
P1.1/UCB0CLK/TA0.1/A1	1	P1.1 (I/O)	I: 0; O: 1	00	0	N/A
		UCB0CLK	X	01	0	N/A
		TA0.CCI1A	0	10	0	N/A
		TA0.1	1			
		A1	X	X	1 (x = 1)	N/A
P1.2/UCB0SIMO/UCB0SDA/TA0.2/A2		P1.2 (I/O)	I: 0; O: 1	00	0	N/A
		UCB0SIMO/UCB0SDA	X	01	0	N/A
		TA0.CCI2A	0	10	0	N/A
		TA0.2	1			
		A2/Veref-	X	X	1 (x = 2)	N/A



```c

int main(void)
{
    WDTCTL = WDTPW | WDTHOLD;                       // Stop watchdog timer

    // Configure GPIO Setup
    // RED LED
    P1DIR |= BIT0;                                  // Set P1.0 as output
    P1OUT |= BIT0;                                  // P1.0 high

    // TA0.CCI2A input capture Setup
    P1SEL1 |= BIT2;                                 // TA0.CCI2A input capture pin, second function
    P1REN |= BIT2;                                  // enable internal pull-down resistor
    P1OUT &= ~BIT2;

    // ACLK Output Setup  Got to externally jumper P2.2 to P1.2
    P2SEL1 |= BIT2;                                 // Set as ACLK pin, second function
    P2DIR |= BIT2;

    // Disable the GPIO power-on default high-impedance mode to activate
    // previously configured port settings
    PM5CTL0 &= ~LOCKLPM5;

```

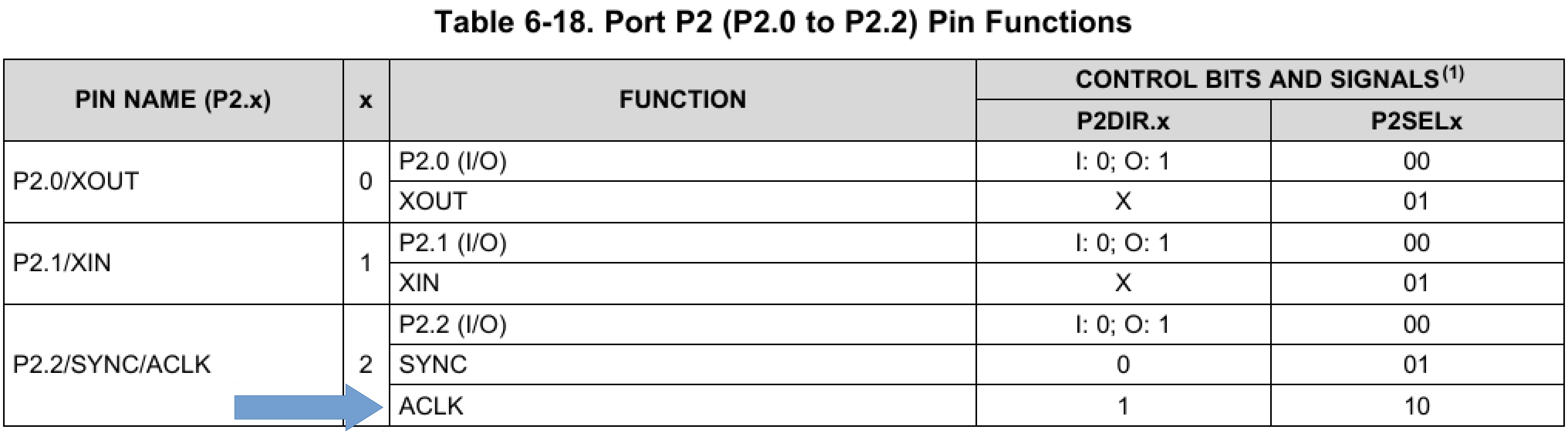
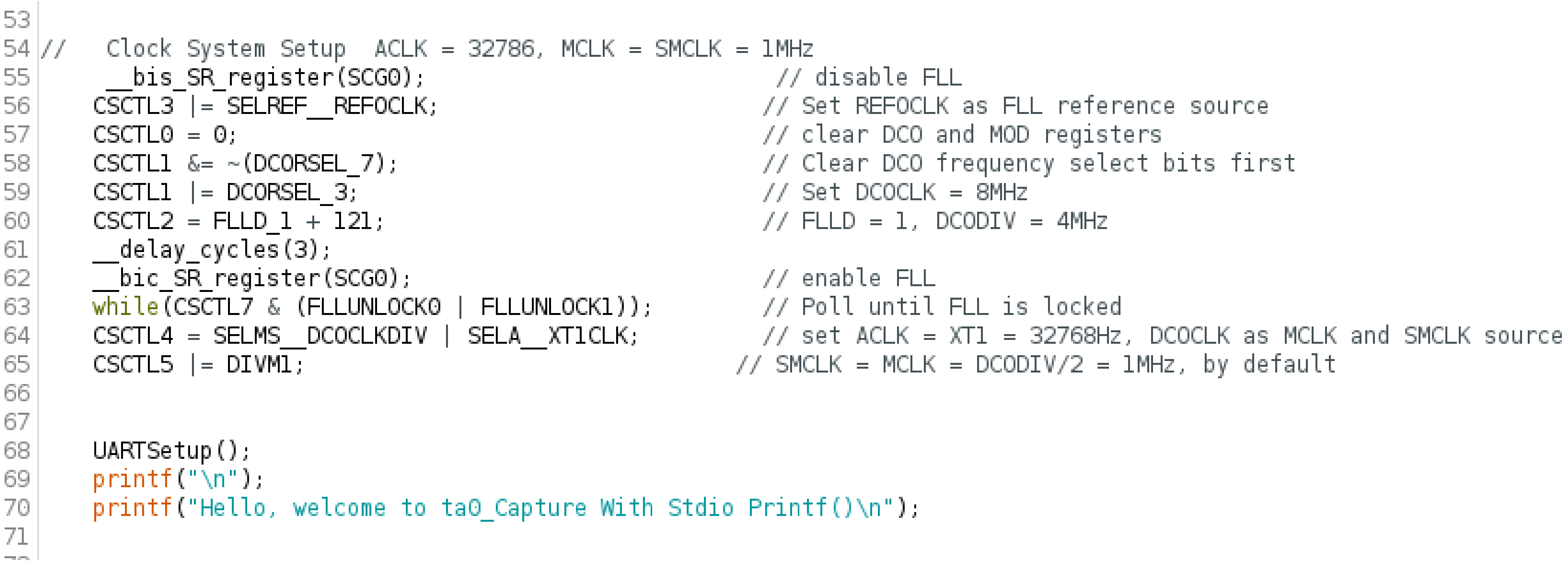
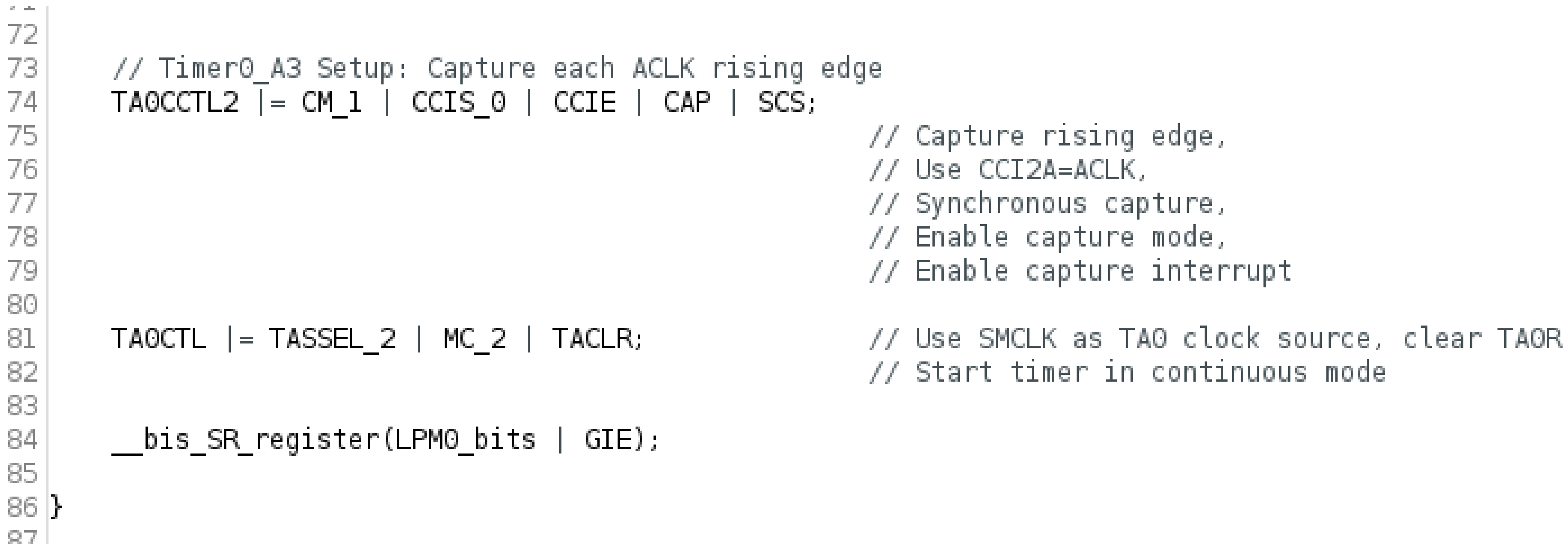


Table 6-18. Port P2 (P2.0 to P2.2) Pin Functions

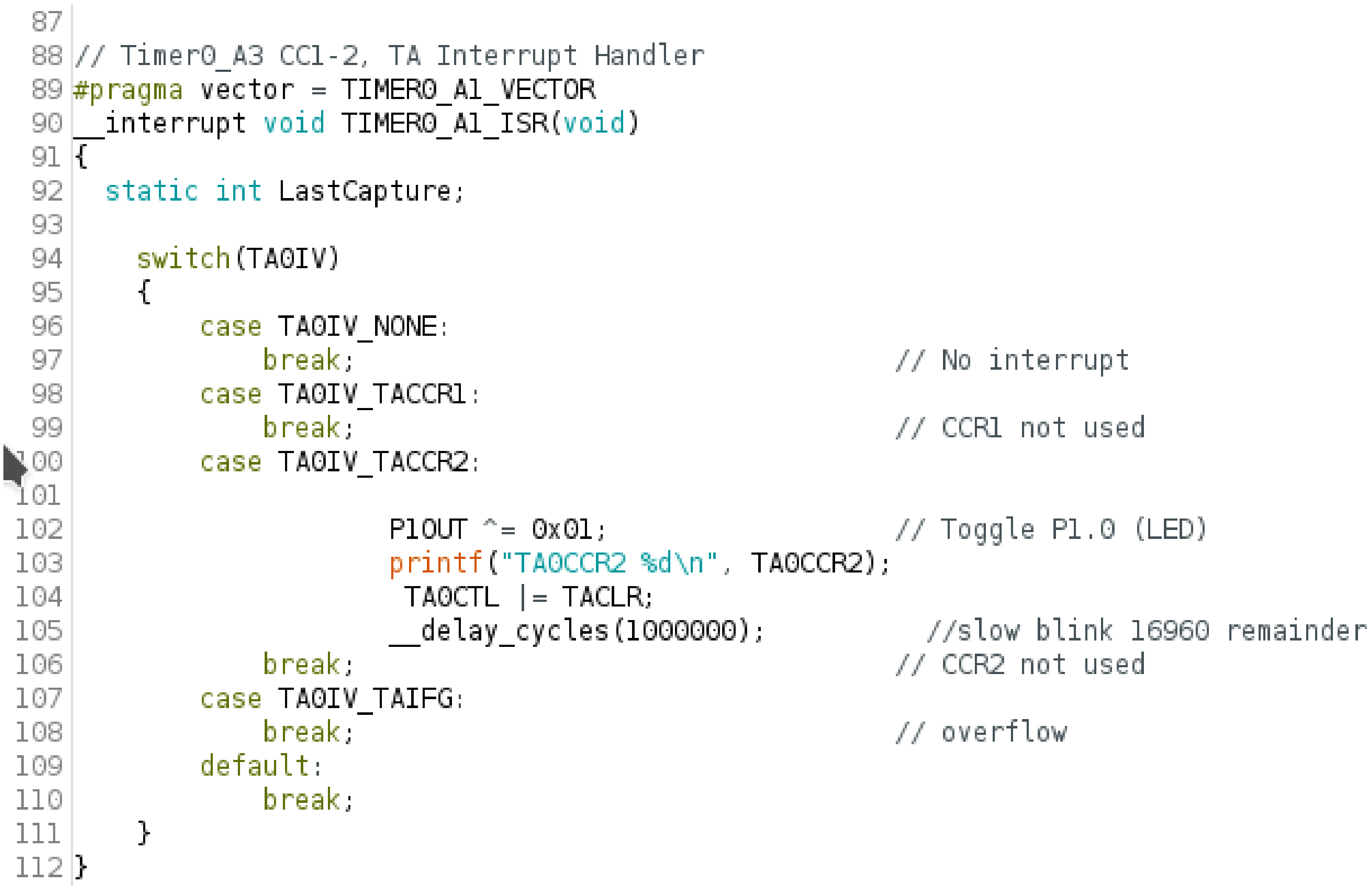
PIN NAME (P2.x)	x	FUNCTION	CONTROL BITS AND SIGNALS[(1)]	
			P2DIR.x	P2SELx
P2.0/XOUT	0	P2.0 (I/O)	I: 0; O: 1	00
		XOUT	X	01
P2.1/XIN	1	P2.1 (I/O)	I: 0; O: 1	00
		XIN	X	01
P2.2/SYNC/ACLK	2	P2.2 (I/O)	I: 0; O: 1	00
		SYNC	0	01
		ACLK	1	10



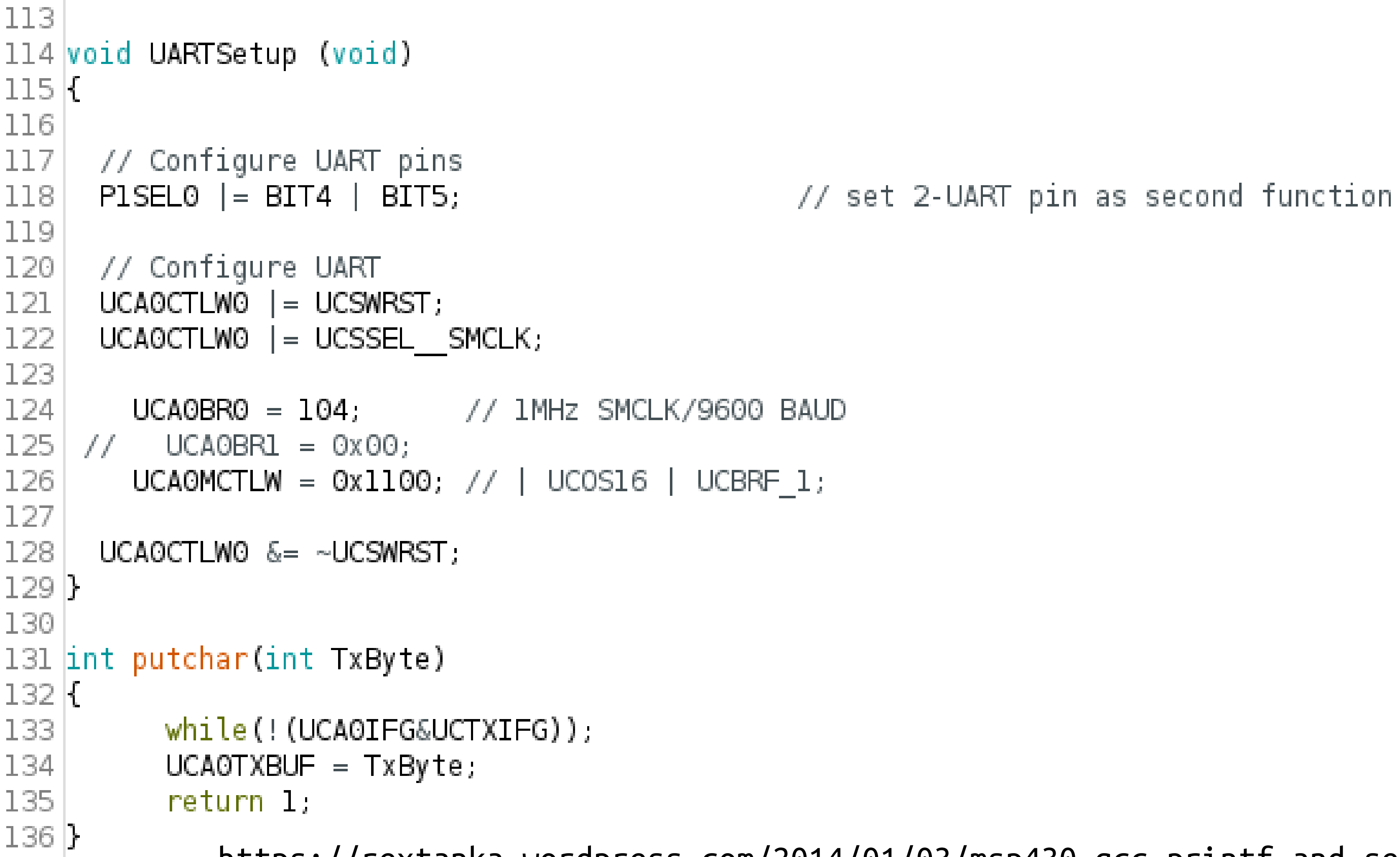
```

//   Clock System Setup  ACLK = 32786, MCLK = SMCLK = 1MHz
     __bis_SR_register(SCG0);                          // disable FLL
    CSCTL3 |= SELREF__REFOCLK;                        // Set REFOCLK as FLL reference source
    CSCTL0 = 0;                                       // clear DCO and MOD registers
    CSCTL1 &= ~(DCORSEL_7);                           // Clear DCO frequency select bits first
    CSCTL1 |= DCORSEL_3;                              // Set DCOCLK = 8MHz
    CSCTL2 = FLLD_1 + 121;                            // FLLD = 1, DCODIV = 4MHz
    __delay_cycles(3);
    __bic_SR_register(SCG0);                          // enable FLL
    while(CSCTL7 & (FLLUNLOCK0 | FLLUNLOCK1));        // Poll until FLL is locked
    CSCTL4 = SELMS__DCOCLKDIV | SELA__XT1CLK;         // set ACLK = XT1 = 32768Hz, DCOCLK as MCLK and SMCLK source
    CSCTL5 |= DIVM1;                                // SMCLK = MCLK = DCODIV/2 = 1MHz, by default


    UARTSetup();
    printf("\n");
    printf("Hello, welcome to ta0_Capture With Stdio Printf()\n");

```



```c

    // Timer0_A3 Setup: Capture each ACLK rising edge
    TA0CCTL2 |= CM_1 | CCIS_0 | CCIE | CAP | SCS;
                                                    // Capture rising edge,
                                                    // Use CCI2A=ACLK,
                                                    // Synchronous capture,
                                                    // Enable capture mode,
                                                    // Enable capture interrupt

    TA0CTL |= TASSEL_2 | MC_2 | TACLR;              // Use SMCLK as TA0 clock source, clear TA0R
                                                    // Start timer in continuous mode

    __bis_SR_register(LPM0_bits | GIE);

}
```



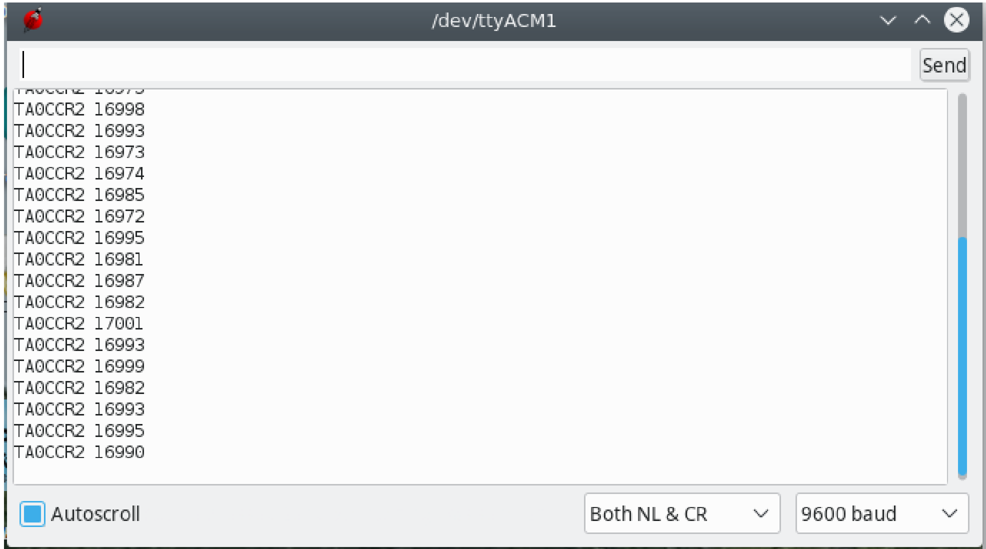
```
// Timer0_A3 CC1-2, TA Interrupt Handler
#pragma vector = TIMER0_A1_VECTOR
__interrupt void TIMER0_A1_ISR(void)
{
  static int LastCapture;

    switch(TA0IV)
    {
        case TA0IV_NONE:
            break;                                  // No interrupt
        case TA0IV_TACCR1:
            break;                                  // CCR1 not used
        case TA0IV_TACCR2:

                    P1OUT ^= 0x01;                  // Toggle P1.0 (LED)
                    printf("TA0CCR2 %d\n", TA0CCR2);
                     TA0CTL |= TACLR;
                    __delay_cycles(1000000);          //slow blink 16960 remainder
            break;                                  // CCR2 not used
        case TA0IV_TAIFG:
            break;                                  // overflow
        default:
            break;
    }
}
```



```c

void UARTSetup (void)
{

 // Configure UART pins
 P1SEL0 |= BIT4 | BIT5;                    // set 2-UART pin as second function

 // Configure UART
 UCA0CTLW0 |= UCSWRST;
 UCA0CTLW0 |= UCSSEL__SMCLK;

   UCA0BR0 = 104;      // 1MHz SMCLK/9600 BAUD
 //   UCA0BR1 = 0x00;
   UCA0MCTLW = 0x1100; // | UCOS16 | UCBRF_1;

 UCA0CTLW0 &= ~UCSWRST;
}

int putchar(int TxByte)
{
      while(!(UCA0IFG&UCTXIFG));
      UCA0TXBUF = TxByte;
      return 1;
}
```

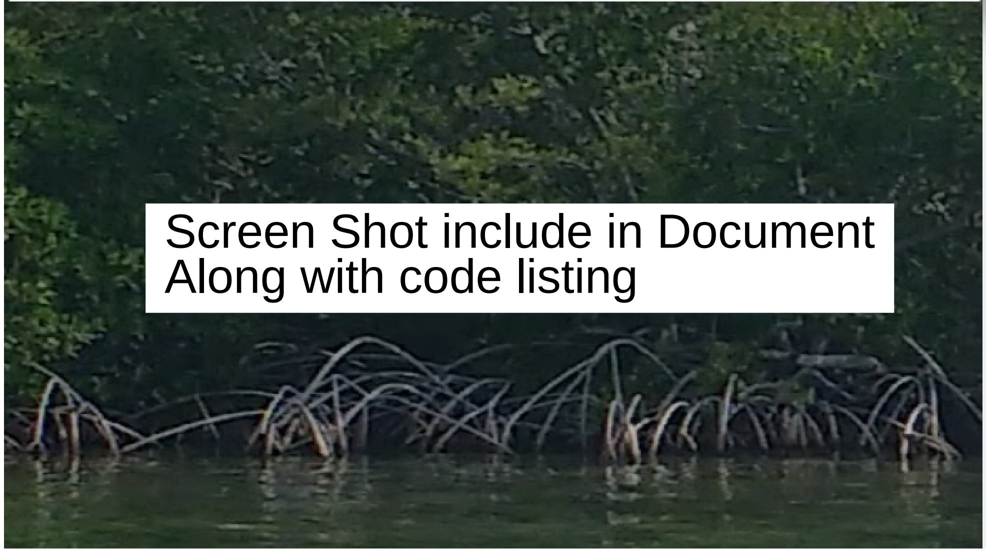
https://rextanka.wordpress.com/2014/01/03/msp430-gcc-printf-and-serial-io/

http://www.msp430launchpad.com/2012/06/using-printf.html

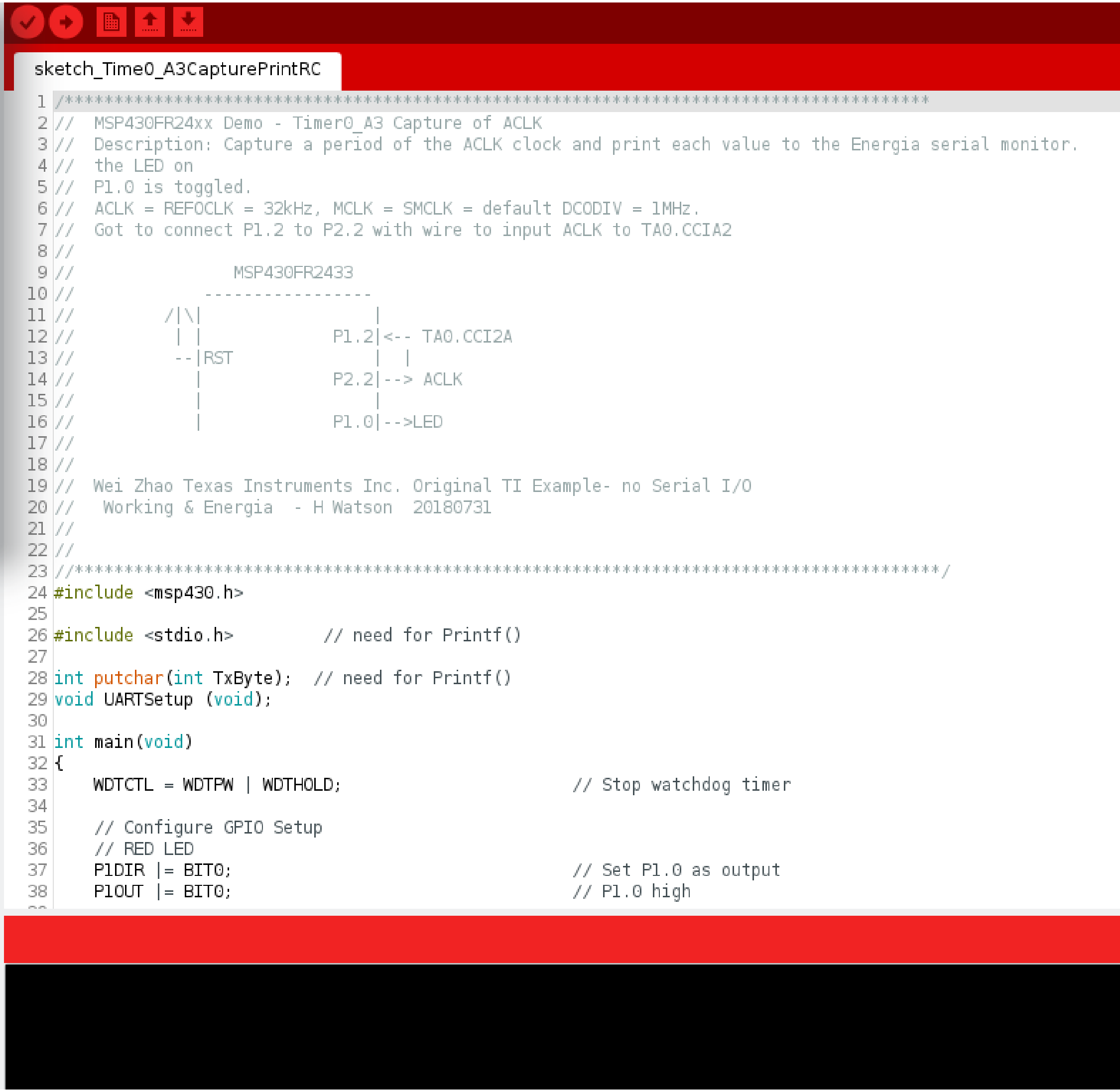


```
TA0CCR2 16975
TA0CCR2 16998
TA0CCR2 16993
TA0CCR2 16973
TA0CCR2 16974
TA0CCR2 16985
TA0CCR2 16972
TA0CCR2 16995
TA0CCR2 16981
TA0CCR2 16987
TA0CCR2 16982
TA0CCR2 17001
TA0CCR2 16993
TA0CCR2 16999
TA0CCR2 16982
TA0CCR2 16993
TA0CCR2 16995
TA0CCR2 16990
```

☑ Autoscroll   Both NL & CR   9600 baud



Screen Shot include in Document
Along with code listing



```
/**************************************************************************************
//  MSP430FR24xx Demo - Timer0_A3 Capture of ACLK
//  Description: Capture a period of the ACLK clock and print each value to the Energia serial monitor.
//  the LED on
//  P1.0 is toggled.
//  ACLK = REFOCLK = 32kHz, MCLK = SMCLK = default DCODIV = 1MHz.
//  Got to connect P1.2 to P2.2 with wire to input ACLK to TA0.CCI2A
//
//                MSP430FR2433
//             -----------------
//         /|\|                 |
//          | |             P1.2|<-- TA0.CCI2A
//          --|RST              |  |
//            |             P2.2|--> ACLK
//            |                 |
//            |             P1.0|-->LED
//
//
//  Wei Zhao Texas Instruments Inc. Original TI Example- no Serial I/O
//   Working & Energia  - H Watson  20180731
//
//
//***************************************************************************************/
#include <msp430.h>

#include <stdio.h>         // need for Printf()

int putchar(int TxByte);  // need for Printf()
void UARTSetup (void);

int main(void)
{
    WDTCTL = WDTPW | WDTHOLD;                       // Stop watchdog timer

    // Configure GPIO Setup
    // RED LED
    P1DIR |= BIT0;                                  // Set P1.0 as output
    P1OUT |= BIT0;                                  // P1.0 high
```