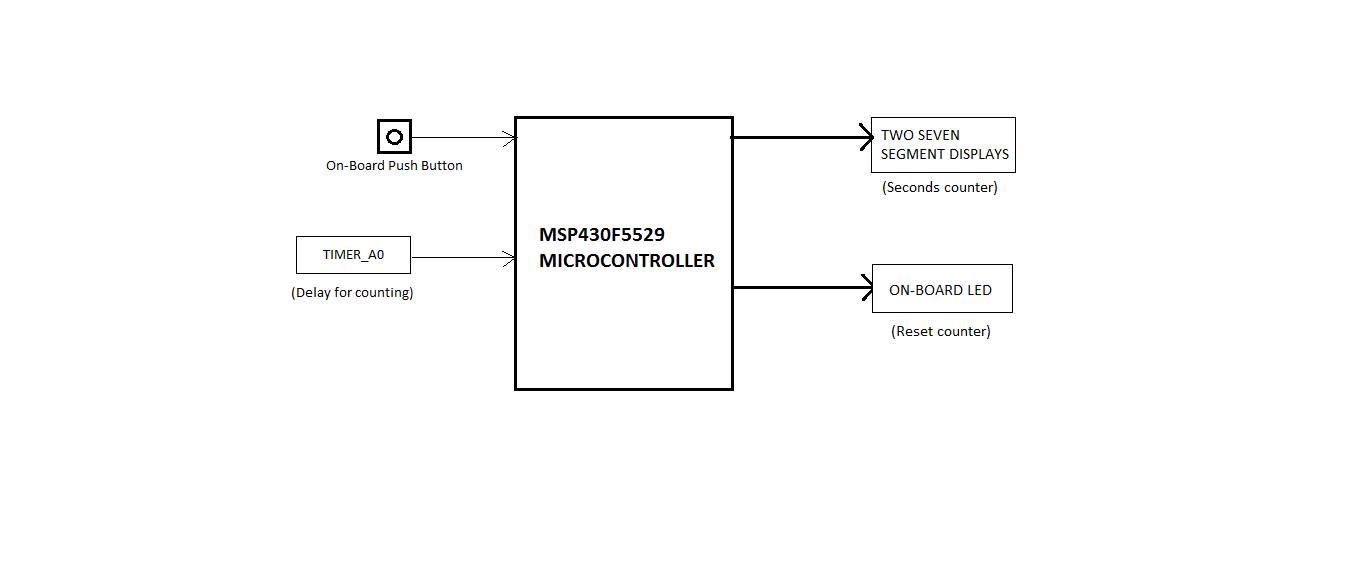
**“60 Seconds Counter using MSP430”**

**Introduction:**

The 60 seconds counter is one which repeatedly counts from 00 to 59 with a gap of 1 second between each count. Pressing a switch serves as a hardware interrupt and resets the count to 00 immediately. An LED is dedicated to count the number of resets by blinking that many number of times i.e the number of times the switch was pressed.

**Block Diagram:**



* On-Board push button is available on pin 2.1. This is used as a hardware interrupt to reset the counter.
* On-Board LED available at pin 1.0 is used to display the number of times reset occurs.
* The two common cathode seven segment displays are connected to port 3 and port 6 and display the seconds count i.e from 00 to 59.
* Timer A is an on-chip peripheral used here to generate delay between each count.

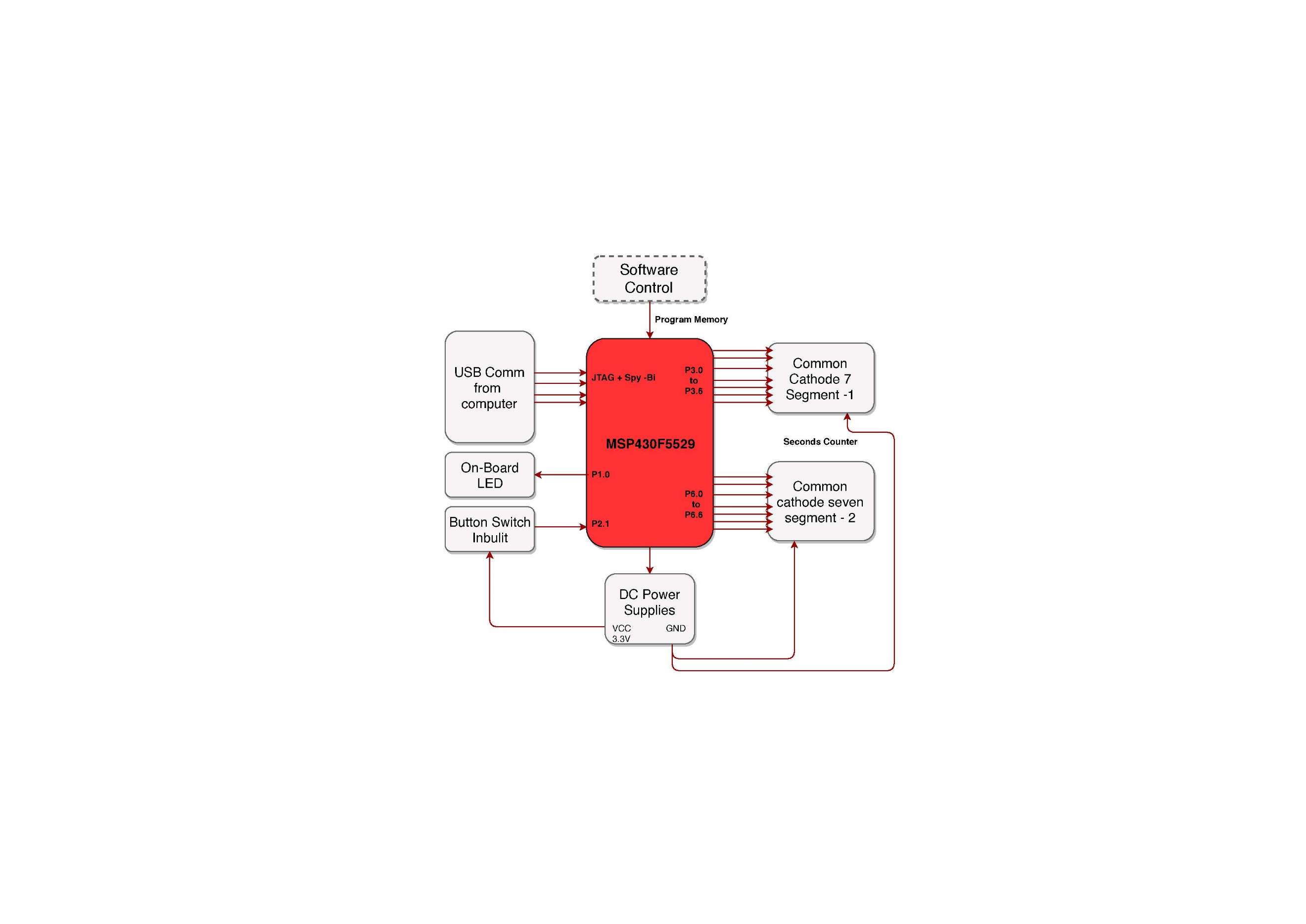
**Specifications and tool used:**

MSP430F5529:

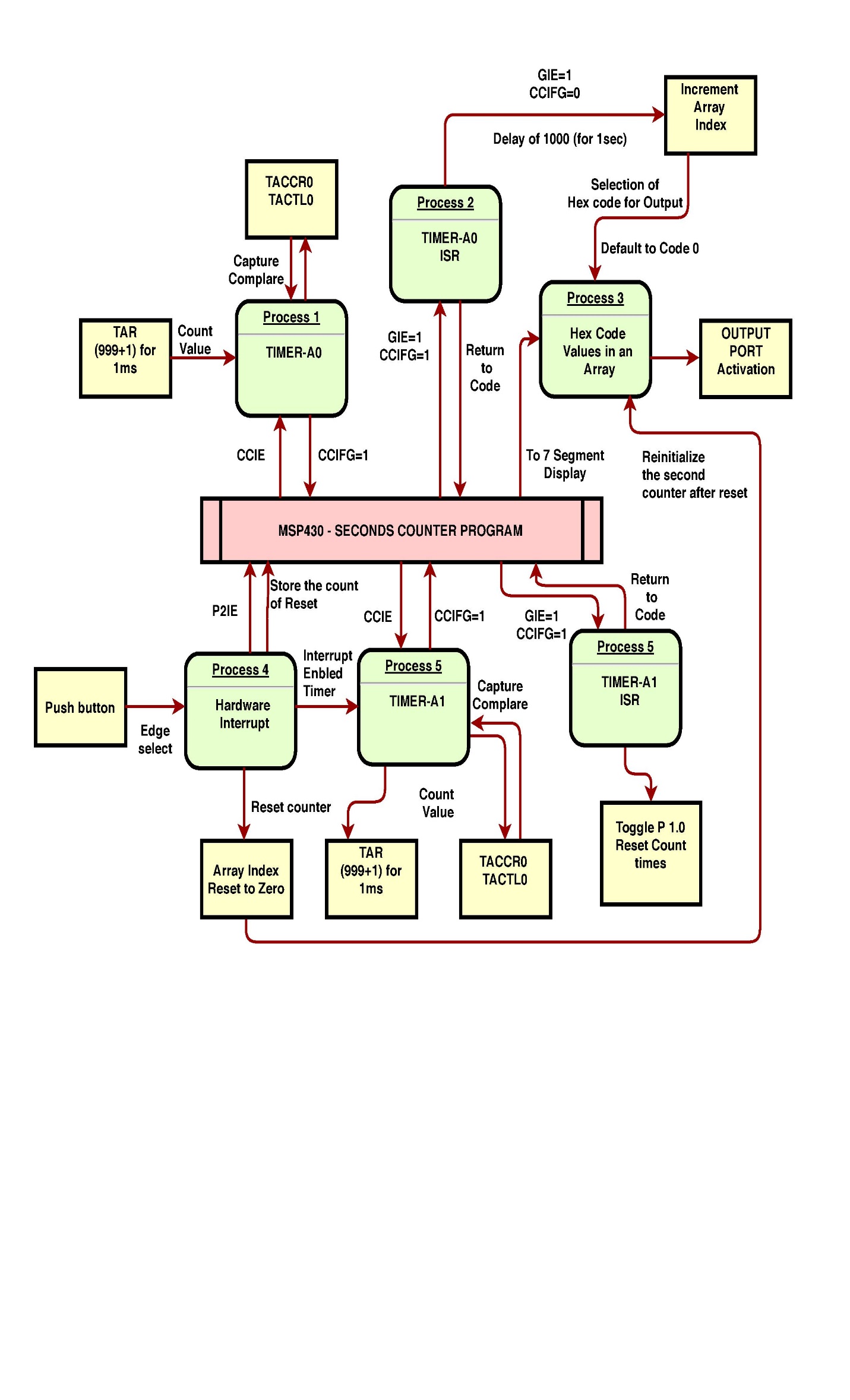
* USB 2.0-enabled MSP430F5529 16-bit MCU
* Has 8 ports.
* Up to 25 MHz clock speed.
* 128KB Flash and 8KB RAM
* 12 Bit SAR ADC
* Various USB device class examples and embedded software libraries available (CDC, HID, MSC)
* eZ-FET lite: Open source onboard debugger with application UART
* One USB connection for emulator and target via the use of an onboard USB hub
* USB as power source: 5V and 3.3V through a high efficiency DC/DC converter
* 40 pin LaunchPad standard leveraging the BoosterPack ecosystem

TOOL USED : CCStudio-Version 8.

**Circuit Diagram:**



**Flow chart:**



**Program with comments:**

**#include** <msp430.h> //include the header files for the msp430 family

**#define** delay 1000 // define a variable delay = 1000

**char** code[]={0xFC,0x60,0xDA,0xF2,0x66,0xB6,0xBE,0xE0,0xFE,0xE6};

**char** code1[]={0x7E,0x30,0x6D,0x79,0x33,0x5B}; //HEX equivalent of BCD numbers

**volatile** **unsigned** **int** x=0,y=0,z=0;

**volatile** **unsigned** **int** v=0,i=0;

/\*\*

\* main.c

\*/

**void** **main**()

{

WDTCTL = WDTPW | WDTHOLD; // stop watchdog timer

P7DIR |= 0xFF; // initialisation of all the ports

P7OUT |= 0x00;

P8DIR |= 0xFF;

P8OUT |= 0x00;

P4DIR |= 0xFF;

P4OUT |= 0x00;

P5DIR |= 0xFF;

P5OUT |= 0x00;

P1DIR=0xFF; //Port 1 declared as output port

P3DIR=0xFF; //Port 3 declared as output port

P6DIR=0xFF; //Port 6 declared as output port

P2DIR=0x00; //Port 2 is made as input port

P2REN=0x02; //Pull up resistor for p 2.1

P2OUT=0x02;

P2IE |=BIT1 ; //Port 2 interrupt

P2IES |=BIT1 ;

P2IFG &= ~BIT1 ;

TA0CCTL0=CCIE; //Timer A0 configured in Capture Compare Mode

TA0CCR0=999; //Count 1000 i.e 1 second

TA0CTL = TASSEL\_2 + MC\_1;

\_BIS\_SR(LPM0\_bits+GIE);

}

// Timer A0 interrupt service routine

**#pragma** vector=TIMER0\_A0\_VECTOR

**\_\_interrupt** **void** **Timer\_A** (**void**) //Interrupt service Routine for TIMER A Interrupt

{

z++;

**if**(z>delay) //after 1 second i.e z>1000

{

P3OUT=code[x];

P6OUT=code1[y];

x++;

**if**(x==10) //First seven segment counts from 0 to 9

{

x=0;

y++;

}

**if**(y==6) //Second seven segment counts from 0 to 6

y=0;

z=0;

}

}

**#pragma** vector=PORT2\_VECTOR //p 2.1 push button is used as hardware interrupt

**\_\_interrupt** **void** **port\_2**(**void**) // ISR for hardware interrupt

{

P2IFG &=~BIT1 ; //PORT 2 interrupt flag bit

x=0; //Reset the counter

y=0;

P3OUT=code[x];

P6OUT=code1[y];

v++; //Variable to count the number of resets

**for**(i=0;i<v;i++)

{ //Blink the LED the number of times reset occurs

P1OUT |= BIT0; // P1.0 = toggle

**\_\_delay\_cycles**(1048576);

P1OUT &=~BIT0; // P1.0 = toggle

**\_\_delay\_cycles**(1048576);

}

}

**Conclusion and Future scope:**

1. Digital watch can be implemented through this process by using Real Time clock along with timers

2. Multiple Timers implementation can be done.

3. Use of Hex decoder IC can be done, to multiplex the pins and reduce the usage of ports.

4. Switch debouncing can be implemented by software debouncing .