# ISP vs IAP

- To program microcontroller (i.e. to store code in ROM/Flash).

## ISP (In System Programming)

- Programming microcontroller IC (burning program in its flash) while keeping it in end-user board, is called as ISP.
- In LPC1768, If pin P2.10 is low while reset, controller enters in bootloader mode. At this time the bootloader code (given by manufacturer in Boot ROM of 8 kb located at 0x1FFF0000) is activated. It takes user program from UART0 and write into the flash at address 0x00000000.

## IAP (In Application Programming)

- Implementing a program that takes user code and write it into the flash is called as IAP. The program can decide from where the code should be written into flash.

### Blueboard USB BL

- In our Blueboard, USB BL is present at location 0x00000000 in flash.
- Upon startup USB BL is executed that blinks LED (P1.29) for three seconds and poll pin P2.10.
- If user press P2.10, it emulates LPC1768 flash as USB storage device. When user writes user code on that USB device, it will be written into the flash at location 0x00002000 (IAP).
- If user doesn't press P2.10, it will continue to execute user located at 0x00002000. Before that it will set VTOR to 0x00002000, so that user code vector table is activated.

# SysTick Timer

- Mainly used to generate interrupt periodically (1 ms to 10 ms usually). This timer interrupt is useful for OS (CPU scheduling).
- SysTick is part of CM3 core, so that porting OS code become easy.
- It is 24-bit down counter and when counter reach zero, interrupt/exception is generated. In vector table its handler is located in slot15.

## SysTick registers

- STRELOAD (24-bit)
    - Initial counter value from where count down begin.
    - After interrupt, this value is reloaded in timer.
- STCURR (24-bit)
    - Timer counter.
    - When it reach to zero, SysTick interrupt is generated.

- STCTRL
    - bit0: (EN) Enable(1) or Disable(0)
    - bit1: (INTREN) interrupt Enable(1) or Disable(0)
    - bit2: (CCLKSRC) Clock source: CCLK(1) or External STCLK(0)
    - bit16: (CNTFLAG) interrupt flag: Set to 1 when counter reach 0. Cleared by reading.

## SysTick config

- cnt = ms * CCLK/1000
- For example:
    - if CCLK=72MHz and need interrupt after every 1 ms.
        - cnt = 1 * 72000000 / 1000 = 72000
    - if CCLK=72MHz and need interrupt after every 10 ms.
        - cnt = 10 * 72000000 / 1000 = 720000

```c
void systick_init(uint32_t ms)
{
    uint32_t cnt = (SystemCoreClock / 1000) * ms;
    STRELOAD = cnt - 1;
    STCURR = 0;
    STCTRL = BV(CCLKSRC) | BV(INTREN) | BV(EN);
}

static volatile int delay_cnt = 0;

void SysTick_Handler(void)
{
    delay_cnt--;
}

void systick_delay_ms(uint32_t ms)
{
    delay_cnt = ms - 1;
    while(delay_cnt > 0)
        ;
}

int main()
{
    systick_init(1); // intr after every 1 ms.
    led_init();
    while(1)
    {
        led_on();
        systick_delay_ms(1000);
        led_off();
        systick_delay_ms(1000);
    }
    return 0;
}
```

```c
void systick_init(uint32_t ms)
{
    uint32_t cnt = (SystemCoreClock / 1000) * ms;
    STRELOAD = cnt - 1;
    STCURR = 0;
    STCTRL = BV(CCLKSRC) | BV(INTREN) | BV(EN);
}

static volatile int jiffies = 0;

void SysTick_Handler(void)
{
    jiffies++;
}

void systick_delay_ms(uint32_t ms)
{
    uint32_t cur = jiffies;
    while(jiffies <  cur + ms)
        ;
}

int main()
{
    systick_init(1); // intr after every 1 ms.
    led_init();
    while(1)
    {
        led_on();
        systick_delay_ms(1000);
        led_off();
        systick_delay_ms(1000);
    }
    return 0;
}
```

# I2C

- I2C stands for Inter IC Communication. It is also called as TWI i.e. Two Wire Interface.
- Developed by Philips for short distance Communication.

## I2C Bus protocol

- I2C is two wire multi-master bus protocol.
- Multiple devices on bus are capable of being master, but only one can be master at a time.
- Master is responsible for generating clock and initiating communication.
- On one bus, max 120 devices can be connected.

## Physical characteristics

- Two wires i.e. SDA and SCL
- They are connected to VCC through 4.7K resistors. They form Wired-AND.
    - If any device pulls line low, effective line voltage is low.
- It follows TTL voltage levels.
- I2C frequency
    - 100 KHz: Standard mode
    - 400 KHz: Fast mode
    - 1 MHz: Fast mode plus
- Half duplex protocol

# Logical characteristics

- Bit change
    - SDA can change only if SCL is low.
- Start bit
    - Falling edge on SDA, while SCL is high.
- Stop bit
    - Rising edge on SDA, while SCL is high.
- Data Frame
    - 8 Data bits (tx) + 1 Ack' bit (rx)
- Address Frame
    - 7 Addr bits (tx-M) + 1 read-write' bit (tx-M) + 1 Ack' bit (rx-S)
    - It is always sent by the master.
    - It is always immediately after START bit.
- Each I2C device have 7-bit Standard I2C address. It is mentioned in datasheet of that device.

## I2C Device modes

- Master Transmitter (MT)
- Master Receiver (MR)
- Slave Transmitter (ST)
- Slave Receiver (SR)

## Single Byte write

- START [MT] + SlaveAddr(W) [MT] + Ack [SR] + Data [MT] + Ack [SR] + STOP [MT]

## Single Byte read

- START [MT] + SlaveAddr(R) [MT] + Ack [SR] + Data [ST] + Ack [MR] + STOP [MT]

### Multi Byte write

- START [MT] + SlaveAddr(W) [MT] + Ack [SR] + InternalAddr [MT] + Ack [SR] + DATA1 [MT] + Ack [SR] + DATA2 [MT] + Ack [SR] + ... + DATAn [MT] + ACK [SR] + STOP [MT]

### Multi Byte readq

- START [MT] + SlaveAddr(W) [MT] + Ack [SR] + InternalAddr [MT] + Ack [SR] + START [MT] + SlaveAddr(R) [MT] + Ack [SR] + DATA1 [ST] + Ack [MR] + DATA2 [ST] + Ack [MR] + ... + DATAn [ST] + nACK [MR] + STOP [MT]

# I2C Applications

- I2C Devices LCD, EEPROM, RTC, ...