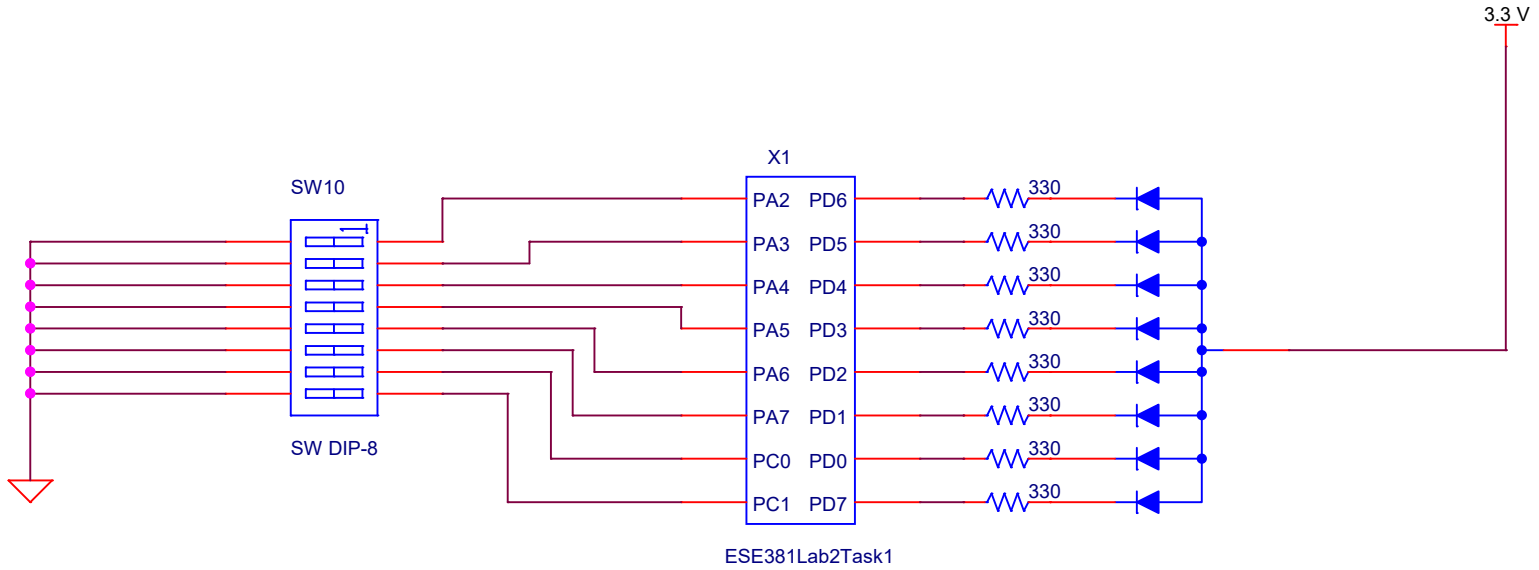
STONY BROOK UNIVERSITY
DEPARTMENT OF COMPUTER AND ELECTRICAL ENGINEERING

ESE 381.L02

Lab 2: Bitwise Logical Operations in C

Name: Jason Tan
SBU ID #: 112319102
Due Date: 11 Feb. 2022 by the end of Lab



Title	ESE 381 Lab 2 Task 1 Schematic	
Size A	Document Number <Doc>	Rev <RevCo
Date: Monday, February 07, 2022	Sheet 1 of 1	

```c
/*
 * parallel_in_parallel_out_flat.c
 *
 * Created: 1/28/2022 9:58:44 PM
 * Author : Jason
 *
 */

#include <avr/io.h>


int main(void)
{
    //pointer to PIN2CTRL array of pin configuration registers
    uint8_t* ptr = (uint8_t*)&PORTA.PIN2CTRL;
    PORTC_PIN0CTRL = 0x08;  //Enable internal pull up for PC0
    PORTC_PIN1CTRL = 0x08;  //Enable internal pull up for PC1

    //DIR is what configure the port pins as inputs or outputs
    VPORTA_DIR = 0x00;  //Configure PORTA pins ( PA7, PA6, PA5, PA4, PA3, PA2) as the
      inputs
    VPORTC_DIR = 0x00;   //Configure PORTC pins ( PC1, PC0) as the inputs
    VPORTD_DIR = 0xFF;   //Configure PORD pins (PD07 through PD00) as output pins

    //Configure PA7 - PA2 as input buffers with internal pull up resistors
    for(uint8_t i = 0; i < 8; i++){
        *(ptr + i) |= PORT_PULLUPEN_bm;
    }


    while (1)
    {

        VPORTD_OUT = ~((VPORTA_IN & 0xFC) | (VPORTC_IN & 0x03) );  //For the purpose
          of reading for PA7-PA2 and for PC1-PC0 and negating when
                                                                   //changing value
                    bit when switch is on and off

    }
}
```

```c
/*
 * parallel_in_parallel_out_struct.c
 *
 * Created: 2/3/2022 10:39:54 PM
 * Author : Jason Tan
 */

#include <avr/io.h>


int main(void)
{
    //pointer to PIN2CTRL array of pin configuration registers
    uint8_t* ptr = (uint8_t*)&PORTA.PIN2CTRL;
    PORTC.PIN0CTRL = 0x08;  //Enable internal pull up for PC0
    PORTC.PIN1CTRL = 0x08;  //Enable internal pull up for PC1

    //DIR is what configure the port pins as inputs or outputs
    VPORTA.DIR = 0x00;  //Configure PORTA pins ( PA7, PA6, PA5, PA4, PA3, PA2) as the
      inputs
    VPORTC.DIR = 0x00;   //Configure PORTC pins ( PC1, PC0) as the inputs
    VPORTD.DIR = 0xFF;   //Configure PORD pins (PD07 through PD00) as output pins

    //Configure PA7 - PA2 as input buffers with internal pull up resistors
    for(uint8_t i = 0; i < 8; i++){
        *(ptr + i) |= PORT_PULLUPEN_bm;
    }


    while (1)
    {
        VPORTD.OUT = ~((VPORTA_IN & 0xFC) | (VPORTC_IN & 0x03));
    }
}
```

```c
/*
 * read_modify_write_sftw_sw0.c
 *
 * Created: 2/3/2022 11:49:03 PM
 * Author : Jason Tan
 */

#include <avr/io.h>


int main(void)
{

    //pointer to PIN2CTRL array of pin configuration registers
    uint8_t* ptr = (uint8_t*)&PORTA.PIN2CTRL;
    PORTC_PIN0CTRL = 0x08;  //Enable internal pull up for PC0
    PORTC_PIN1CTRL = 0x08;  //Enable internal pull up for PC1

    //DIR is what configure the port pins as inputs or outputs
    VPORTA_DIR = 0x00;  //Configure PORTA pins ( PA7, PA6, PA5, PA4, PA3, PA2) as the inputs
    VPORTC_DIR = 0x00;   //Configure PORTC pins ( PC1, PC0) as the inputs
    VPORTD_DIR = 0xFF;   //Configure PORTD pins (PD07 through PD00) as output pins

    VPORTB_DIR = 0x08;   //PB3 output for LED0
    PORTB_PIN2CTRL = 0x08;  //Pull up enable for SW0

    //Configure PA7 - PA2 as input buffers with internal pull up resistors
    for(uint8_t i = 0; i < 8; i++){
        *(ptr + i) |= PORT_PULLUPEN_bm;
    }



    uint8_t n = 3;  //Field value starting from 3
    uint8_t field_mask = 0x0F; //Field mask
    uint8_t field_val; //Read PA3-PA2 and PC1-PC0.
    VPORTD_OUT = ~((VPORTA_IN & 0xFC) | (VPORTC_IN & 0x03));

    while (1)
    {
        //Check for if SW0 is not pressed meaning that sends a logic 1
        //if((VPORTB_IN & PIN2_bm)){
        //  VPORTD_OUT = ~((VPORTA_IN & 0xFC) | (VPORTC_IN & 0x03));
        //}
        //Check for if SW0 is press meaning that sends a logic 0
        field_val = ((VPORTC_IN & 0x03 ) | (VPORTA_IN & 0x0C));
        if(!(VPORTB_IN & PIN2_bm)){
            //                          0b1000 0111          0b1111 PA3 PA2 PC1 PC0 -> 1 PA3 PA2 PC1 PC0 000
            VPORTD_OUT = (VPORTD_OUT & ~(field_mask << n)) | (((field_val & field_mask) << n) ^ 0x78);
```

```
        }



        /*
        //If PB2 is at logic 0 which means SW0 is pressed
        if(~(VPORTB_IN & PIN2_bm)){

            //When PC0 is pressed which sends logic 0
            if(~(VPORTC_IN & PIN0_bm)){
                PORTD_OUT &= ~(1 << 3); //Clears bit 3 to turn on the LED
            }
            //When PC0 is not pressed
            else if((VPORTC_IN & PIN0_bm)){
                PORTD_OUT |= (1 << 3); //Set bit 3 to turn off the LED
            }


            //When PC1 is pressed which sends logic 0
            if (~(VPORTC_IN & PIN1_bm)){
                PORTD_OUT &= ~(1 << 4); //Clears bit 4 to turn on the LED
            }
            //When PC1 is not pressed
            else if((VPORTC_IN & PIN1_bm)){
                PORTD_OUT |= (1 << 4); //Set bit 4 to turn off the LED
            }

            //When PA2 is pressed which sends logic 0
            if (~(VPORTA_IN & PIN2_bm)){
                PORTD_OUT &= ~(1 << 5); //Clears bit 5 to turn on the LED
            }
            //When PA2 is not pressed
            else if((VPORTA_IN & PIN2_bm)){
                PORTD_OUT |= (1 << 5); //Set bit 5 to turn off the LED
            }


            //When PA3 is pressed which sends logic 0
            if (~(VPORTA_IN & PIN3_bm)){
                PORTD_OUT &= ~(1 << 6); //Clears bit 6 to turn on the LED
            }
            //When PA3 is not pressed
            else if((VPORTA_IN & PIN3_bm)){
                PORTD_OUT |= (1 << 6); //Set bit 6 to turn off the LED
            }
        }

        //If PB2 is at logic 1 which means SW0 not pressed
        else if ((VPORTB_IN & PIN2_bm)){

            //From Lab 2 Task 1
```

```
            //When PC0 is pressed which sends logic 0
            if(~(VPORTC_IN & PIN0_bm)){
                PORTD_OUT &= ~(1 << 0); //Clears bit 0 to turn on the LED
            }
            //When PC0 is not pressed
            else if((VPORTC_IN & PIN0_bm)){
                PORTD_OUT |= (1 << 0); //Set bit 0 to turn off the LED
            }


            //When PC1 is pressed which sends logic 0
            if (~(VPORTC_IN & PIN1_bm)){
                PORTD_OUT &= ~(1 << 1);  //Clears bit 1 to turn on the LED
            }
            //When PC1 is not pressed
            else if((VPORTC_IN & PIN1_bm)){
                PORTD_OUT |= (1 << 1); //Set bit 1 to turn off the LED
            }


            //When PA2 is pressed which sends logic 0
            if (~(VPORTA_IN & PIN2_bm)){
                PORTD_OUT &= ~(1 << 2); //Clears bit 2 to turn on the LED
            }
            //When PA2 is not pressed
            else if((VPORTC_IN & PIN2_bm)){
                PORTD_OUT |= (1 << 2); //Set bit 2 to turn off the LED
            }


            //When PA3 is pressed which sends logic 0
            if (~(VPORTA_IN & PIN3_bm)){
                PORTD_OUT &= ~(1 << 3); //Clears bit 3 to turn on the LED
            }
            //When PA3 is not pressed
            else if((VPORTC_IN & PIN3_bm)){
                PORTD_OUT |= (1 << 3); //Set bit 3 to turn off the LED
            }

            //When PA4 is pressed which sends logic 0
            if (~(VPORTA_IN & PIN4_bm)){
                PORTD_OUT &= ~(1 << 4); //Clears bit 4 to turn on the LED
            }
            //When PA4 is not pressed
            else if((VPORTC_IN & PIN4_bm)){
                PORTD_OUT |= (1 << 4); //Set bit 3 to turn off the LED
            }



            //When PA5 is pressed which sends logic 0
            if (~(VPORTA_IN & PIN5_bm)){
```

```
                PORTD_OUT &= ~(1 << 5); //Clears bit 5 to turn on the LED
            }
            //When PA5 is not pressed
            else if((VPORTC_IN & PIN3_bm)){
                PORTD_OUT |= (1 << 5); //Set bit 5 to turn off the LED
            }


            //When PA6 is pressed which sends logic 0
            if (~(VPORTA_IN & PIN6_bm)){
                PORTD_OUT &= ~(1 << 6); //Clears bit 6 to turn on the LED
            }
            //When PA5 is not pressed
            else if((VPORTC_IN & PIN6_bm)){
                PORTD_OUT |= (1 << 6); //Set bit 6 to turn off the LED
            }

            //When PA7 is pressed which sends logic 0
            if (~(VPORTA_IN & PIN6_bm)){
                PORTD_OUT &= ~(1 << 7); //Clears bit 7 to turn on the LED
            }
            //When PA5 is not pressed
            else if((VPORTC_IN & PIN6_bm)){
                PORTD_OUT |= (1 << 7); //Set bit 7 to turn off the LED
            }
        }

        */

    }
}
```

```c
/*
 * xor3_logic_ops.c
 *
 * Created: 2/6/2022 4:02:10 PM
 * Author : jason
 */

#include <avr/io.h>


int main(void)
{

    //pointer to PIN5CTRL array of pin configuration registers
    uint8_t* ptr = (uint8_t*)&PORTA.PIN2CTRL;

    //DIR is what configure the port pins as inputs or outputs
    VPORTA_DIR = 0x00;  //Configure PORTA pins ( PA7, PA6, PA5) as the inputs

    VPORTD_DIR = PIN7_bm;   //Configure (PD7) as output pin

    /*
    PA7 is "C"
    PA6 is "B"
    PA5 is "A"
    PD7 is "F"
    */

    //Configure PA7 - PA5 as input buffers with internal pull up resistors
    for(uint8_t i = 3; i < 8; i++){
        *(ptr + i) |= PORT_PULLUPEN_bm;
    }

    while (1)
    {
        uint8_t C = PORTA_IN & PIN7_bm;
        uint8_t B = (PORTA_IN & PIN6_bm) << 1;   //Shift left by 1 to bit 6 compare
          the bit 5 position (B) to bit 4 position (A)
        uint8_t A = (PORTA_IN & PIN5_bm) << 2;  //Shift left by 1 to compare the bit 5
           position (B) to bit 4 position (A)
        //Aligns the bit 7, bit 6 and bit 5 positions to compare


        PORTD_OUT = (((~C) ^ (~B) ^ (A)) ^ 0xFF ) | (((~C) ^ (B) ^ (~A)) ^ 0xFF ) |
          ( ((C) ^ (~B) ^ (~A)) ^ 0xFF) | (((C) ^ (B) ^ (A)) ^ 0xFF );

        //if(F){
        //  PORTD_OUT &= ~PIN7_bm;  //Turn on LED if F is 1
        //}
        //else{
        //  PORTD_OUT |= PIN7_bm;   //Turn off LED if F is 0;
        //}
```

```
/*
// If PA7 is "C" = 0 | PA6 is "B" = 0 | PA5 is "A" = 0 | PD7 is "F" = 0
//Meaning PA7: pressed | PA6: pressed | PA5: pressed | PD7: LED off (logic 1)
if((~(VPORTA_IN & PIN7_bm)) && (~(VPORTA_IN & PIN6_bm)) && (~(VPORTA_IN &
  PIN5_bm))){
    PORTD_OUT |= PIN7_bm;  //Set PD7 to 1 to turn off LED (logic 0)
}

// If PA7 is "C" = 0 | PA6 is "B" = 0 | PA5 is "A" = 1 | PD7 is "F" = 1
//Meaning PA7: pressed | PA6: pressed | PA5: not pressed | PD7: LED on (logic
  0)
else if((~(VPORTA_IN & PIN7_bm)) && (~(VPORTA_IN & PIN6_bm)) && ((VPORTA_IN &
  PIN5_bm))){
    PORTD_OUT &= ~PIN7_bm;  //Clear PD7 to 0 to turn off LED (logic 0)
}

// If PA7 is "C" = 0 | PA6 is "B" = 1 | PA5 is "A" = 0 | PD7 is "F" = 1
//Meaning PA7: pressed | PA6: not pressed | PA5: pressed | PD7: LED on (send
  logic 0)
else if((~(VPORTA_IN & PIN7_bm)) && ((VPORTA_IN & PIN6_bm)) && (~(VPORTA_IN &
  PIN5_bm))){
    PORTD_OUT &= ~PIN7_bm;  //Clear PD7 to 0 to turn off LED (logic 0)
}

// If PA7 is "C" = 0 | PA6 is "B" = 1 | PA5 is "A" = 1 | PD7 is "F" = 0
//Meaning PA7: pressed | PA6: not pressed | PA5: not pressed | PD7: LED off
  (send logic 1)
else if((~(VPORTA_IN & PIN7_bm)) && ((VPORTA_IN & PIN6_bm)) && ((VPORTA_IN &
  PIN5_bm))){
    PORTD_OUT |= PIN7_bm;  //Set PD7 to 0 to turn off LED (logic 0)
}

// If PA7 is "C" = 1 | PA6 is "B" = 0 | PA5 is "A" = 0 | PD7 is "F" = 1
//Meaning PA7: not pressed | PA6: pressed | PA5: pressed | PD7: LED on (send
  logic 0)
else if(((VPORTA_IN & PIN7_bm)) && (~(VPORTA_IN & PIN6_bm)) && (~(VPORTA_IN &
  PIN5_bm))){
    PORTD_OUT &= ~PIN7_bm;  //Set PD7 to 0 to turn off LED (logic 0)
}
```

```
        // If PA7 is "C" = 1 | PA6 is "B" = 0 | PA5 is "A" = 1 | PD7 is "F" = 0
        //Meaning PA7: not pressed | PA6: pressed | PA5: not pressed | PD7: LED off
          (send logic 1)
        else if(((VPORTA_IN & PIN7_bm)) && (~(VPORTA_IN & PIN6_bm)) && ((VPORTA_IN &
          PIN5_bm))){
            PORTD_OUT |= PIN7_bm;  //Set PD7 to 0 to turn off LED (logic 0)
        }

        // If PA7 is "C" = 1 | PA6 is "B" = 1 | PA5 is "A" = 0 | PD7 is "F" = 0
        //Meaning PA7: not pressed | PA6: pressed | PA5: not pressed | PD7: LED off
          (send logic 1)
        else if(((VPORTA_IN & PIN7_bm)) && ((VPORTA_IN & PIN6_bm)) && (~(VPORTA_IN &
          PIN5_bm))){
            PORTD_OUT |= PIN7_bm;  //Set PD7 to 1 to turn off LED (logic 0)
        }

        // If PA7 is "C" = 1 | PA6 is "B" = 1 | PA5 is "A" = 1 | PD7 is "F" = 1
        //Meaning PA7: not pressed | PA6: not pressed | PA5: not pressed | PD7: LED on
           (send logic 0)
        else if(((VPORTA_IN & PIN7_bm)) && ((VPORTA_IN & PIN6_bm)) && ((VPORTA_IN &
          PIN5_bm))){
            PORTD_OUT &= ~PIN7_bm;  //Set PD7 to 1 to turn off LED (logic 0)
        }
        */

    }
}
```

```
/*
 * data.h
 *
 * Created: 2/6/2022 4:46:15 PM
 *  Author: Jason
 */


#ifndef DATA_H_
#define DATA_H_

typedef union {

    uint8_t byte;        //member used for register access

    struct{
        /*
        uint8_t C : 1;   //Truth table C column
        uint8_t B : 1;   //Truth table B column
        uint8_t A : 1;   //Truth table A column
        uint8_t F : 1;   //Truth table F column
        */
        uint8_t bit0: 1;  //bit0
        uint8_t bit1: 1;    //bit1
        uint8_t bit2: 1;    //bit2
        uint8_t bit3: 1;//bit3
        uint8_t bit4: 1;//bit4
        uint8_t bit5: 1;//bit5
        uint8_t bit6: 1;//bit6
        uint8_t bit7: 1;//bit7


        }bvals;

    } Named_bits;



#endif /* DATA_H_ */
```

```c
/*
 * xor3_named_bits.c
 *
 * Created: 2/6/2022 4:32:55 PM
 * Author : Jason Tan
 */

#include <avr/io.h>
#include "data.h"


int main(void)
{
    //pointer to PIN5CTRL array of pin configuration registers
    uint8_t* ptr = (uint8_t*)&PORTA.PIN2CTRL;

    //DIR is what configure the port pins as inputs or outputs
    VPORTA_DIR = 0x00;  //Configure PORTA pins ( PA7, PA6, PA5) as the inputs
    VPORTD_DIR = 0x80;   //Configure (PD7) as output pin

    /*
    PA7 is "C"
    PA6 is "B"
    PA5 is "A"
    PD7 is "F"
    */

    //Configure PA7 - PA5 as input buffers with internal pull up resistors
    for(uint8_t i = 3; i < 8; i++){
        *(ptr + i) |= PORT_PULLUPEN_bm;
    }

    Named_bits data;   //Data to be processed

    uint8_t temp_in = 0xFF;  //Initialize the LED to be off
    uint8_t temp_out = 0x00;  //Initialize the temp output to be all 0x00

    while (1)
    {
        uint8_t C = PORTA_IN & PIN7_bm;
        uint8_t B = (PORTA_IN & PIN6_bm) << 1;
        uint8_t A = (PORTA_IN & PIN5_bm) << 2;
    //  temp_in = 0xFF;  //Initialize the LED to be off
    //  data.bvals.bit7 = (VPORTA_IN & PIN7_bm);  //To read in terms of whether the
      switch is on (logic 0) or off (logic 1) for PA7
    //  data.bvals.bit6 = (VPORTA_IN & PIN6_bm);  //To read in terms of whether the
      switch is on (logic 0) or off (logic 1) for PA6
    //  data.bvals.bit5 = (VPORTA_IN & PIN5_bm);  //To read in terms of whether the
      switch is on (logic 0) or off (logic 1) for PA5

    //  data.bvals.C = (VPORTA_IN & PIN7_bm);  //To read in terms of whether the
      switch is on (logic 0) or off (logic 1) for PA7
```

```
    //  data.bvals.B = (VPORTA_IN & PIN6_bm);  //To read in terms of whether the switch is on (logic 0) or off (logic 1) for PA6
    //  data.bvals.A = (VPORTA_IN & PIN5_bm);  //To read in terms of whether the switch is on (logic 0) or off (logic 1) for PA5

        data.byte = temp_in;  //Set that register as the initial output value



        //Canonical sum of products of the 3 input truth table and store that into
        data.byte =  (((~C) ^ (~B) ^ (A)) ^ 0xFF ) | (((~C) ^ (B) ^ (~A)) ^ 0xFF ) | (((C) ^ (~B) ^ (~A)) ^ 0xFF ) | (((C) ^ (B) ^ (A)) ^ 0xFF );

        //Store that bit result after doing the canonical sum of product
        temp_out = data.byte;

        //To output the result
        PORTD_OUT = temp_out;


        // If PA7 is "C" = 0 | PA6 is "B" = 0 | PA5 is "A" = 0 | PD7 is "F" = 0
        //Meaning PA7: pressed | PA6: pressed | PA5: pressed | PD7: LED off (logic 1)
        //if(~(data.bvals.C) && (~(data.bvals.B)) && (~(data.bvals.A))){
            //data.bvals.bit

    //      PORTD_OUT |= PIN7_bm;  //Set PD7 to 1 to turn off LED (logic 0)
        //}

        /*
        // If PA7 is "C" = 0 | PA6 is "B" = 0 | PA5 is "A" = 1 | PD7 is "F" = 1
        //Meaning PA7: pressed | PA6: pressed | PA5: not pressed | PD7: LED on (logic 0)
        else if((~(VPORTA_IN & PIN7_bm)) && (~(VPORTA_IN & PIN6_bm)) && ((VPORTA_IN & PIN5_bm))){
            PORTD_OUT &= ~PIN7_bm;  //Clear PD7 to 0 to turn off LED (logic 0)
        }

        // If PA7 is "C" = 0 | PA6 is "B" = 1 | PA5 is "A" = 0 | PD7 is "F" = 1
        //Meaning PA7: pressed | PA6: not pressed | PA5: pressed | PD7: LED on (send logic 0)
        else if((~(VPORTA_IN & PIN7_bm)) && ((VPORTA_IN & PIN6_bm)) && (~(VPORTA_IN & PIN5_bm))){
            PORTD_OUT &= ~PIN7_bm;  //Clear PD7 to 0 to turn off LED (logic 0)
        }

        // If PA7 is "C" = 0 | PA6 is "B" = 1 | PA5 is "A" = 1 | PD7 is "F" = 0
        //Meaning PA7: pressed | PA6: not pressed | PA5: not pressed | PD7: LED off (send logic 1)
        else if((~(VPORTA_IN & PIN7_bm)) && ((VPORTA_IN & PIN6_bm)) && ((VPORTA_IN & PIN5_bm))){
            PORTD_OUT |= PIN7_bm;  //Set PD7 to 0 to turn off LED (logic 0)
        }
```

```
        // If PA7 is "C" = 1 | PA6 is "B" = 0 | PA5 is "A" = 0 | PD7 is "F" = 1
        //Meaning PA7: not pressed | PA6: pressed | PA5: pressed | PD7: LED on (send
          logic 0)
        else if(((VPORTA_IN & PIN7_bm)) && (~(VPORTA_IN & PIN6_bm)) && (~(VPORTA_IN &
          PIN5_bm))){
            PORTD_OUT &= ~PIN7_bm;  //Set PD7 to 0 to turn off LED (logic 0)
        }

        // If PA7 is "C" = 1 | PA6 is "B" = 0 | PA5 is "A" = 1 | PD7 is "F" = 0
        //Meaning PA7: not pressed | PA6: pressed | PA5: not pressed | PD7: LED off
          (send logic 1)
        else if(((VPORTA_IN & PIN7_bm)) && (~(VPORTA_IN & PIN6_bm)) && ((VPORTA_IN &
          PIN5_bm))){
            PORTD_OUT |= PIN7_bm;  //Set PD7 to 0 to turn off LED (logic 0)
        }

        // If PA7 is "C" = 1 | PA6 is "B" = 1 | PA5 is "A" = 0 | PD7 is "F" = 0
        //Meaning PA7: not pressed | PA6: pressed | PA5: not pressed | PD7: LED off
          (send logic 1)
        else if(((VPORTA_IN & PIN7_bm)) && ((VPORTA_IN & PIN6_bm)) && (~(VPORTA_IN &
          PIN5_bm))){
            PORTD_OUT |= PIN7_bm;  //Set PD7 to 1 to turn off LED (logic 0)
        }

        // If PA7 is "C" = 1 | PA6 is "B" = 1 | PA5 is "A" = 1 | PD7 is "F" = 1
        //Meaning PA7: not pressed | PA6: not pressed | PA5: not pressed | PD7: LED on
           (send logic 0)
        else if(((VPORTA_IN & PIN7_bm)) && ((VPORTA_IN & PIN6_bm)) && ((VPORTA_IN &
          PIN5_bm))){
            PORTD_OUT &= ~PIN7_bm;  //Set PD7 to 1 to turn off LED (logic 0)
        }
        */

    }
}
```