

# Binary Representations, Serial Communication and the Atmel 2560

# Administration...

- Top Hat or Zyante problems?

# Questions?

# Quiz

# Data Types

- short, long, int: size depends on the particular microprocessor
- In order to be clear about sizes, gcc (our compiler) provides a set of types, including:
  - `int8_t` 8-bit signed
  - `uint16_t` 16-bit unsigned
- Use these for our projects!

# Atmel Mega2560 Microcontroller

# Atmel Mega2560

U1IO

90	PF7(ADC7/TDI)	PA7(AD7)	71
91	PF6(ADC6/TDO)	PA6(AD6)	72
92	PF5(ADC5/TMS)	PA5(AD5)	73
93	PF4(ADC4/TCK)	PA4(AD4)	74
94	PF3(ADC3)	PA3(AD3)	75
95	PF2(ADC2)	PA2(AD2)	76
96	PF1(ADC1)	PA1(AD1)	77
97	PF0(ADC0)	PA0(AD0)	78
1	PG5(OC0B)	PB7(OC0A/OC1C)	26
29	PG4(TOSC1)	PB6(OC1B)	25
28	PG3(TOSC2)	PB5(OC1A)	24
70	PG2(ALE)	PB4(OC2A)	23
52	PG1(RD)	PB3(MISO)	22
51	PG0(WR)	PB2(MOSI)	21
27	PH7(T4)	PB1(SCK)	20
18	PH6(OC2B)	PB0(SS)	19
17	PH5(OC4C)	PC7(A15)	60
16	PH4(OC4B)	PC6(A14)	59
15	PH3(OC4A)	PC5(A13)	58
14	PH2(XCK2)	PC4(A12)	57
13	PH1(TXD2)	PC3(A11)	56
12	PH0(RXD2)	PC2(A10)	55
79	PJ7	PC1(A9)	54
69	PJ6(PCINT15)	PC0(A8)	53
68	PJ5(PCINT14)	PD7(T0)	50
67	PJ4(PCINT13)	PD6(T1)	49
66	PJ3(PCINT12)	PD5(XCK1)	48
65	PJ2(XCK3)	PD4(ICP1)	47
64	PJ1(TXD3)	PD3(TXD1/INT3)	46
63	PJ0(RXD3)	PD2(RXD1/INT2)	45
82	PK7(ADC15)	PD1(SDA/INT1)	44
83	PK6(ADC14)	PD0(SCL/INT0)	43
84	PK5(ADC13)	PE7(ICP3/INT7)	9
85	PK4(ADC12)	PE6(T3/INT6)	8
86	PK3(ADC11)	PE5(OC3C/INT5)	7
87	PK2(ADC10)	PE4(OC3B/INT4)	6
88	PK1(ADC9)	PE3(OC3A/AIN1)	5
89	PK0(ADC8)	PE2(XCK0/AIN0)	4
42		PE1(TXD0)	3
41	PL7	PE0(RXD0)	2
40	PL6		
39	PL5(OC5C)		
38	PL4(OC5B)		
37	PL3(OC5A)		
36	PL2(T5)		
35	PL1(ICP5)		
	PL0(ICP4)		

# Atmel Mega2560

Pins are organized  
into 8-bit “Ports”:

- A, B, C ... L
  - But no “I”

U1IO	
90	PF7(ADC7/TDI)
91	PF6(ADC6/TDO)
92	PF5(ADC5/TMS)
93	PF4(ADC4/TCK)
94	PF3(ADC3)
95	PF2(ADC2)
96	PF1(ADC1)
97	PF0(ADC0)
1	PG5(OC0B)
29	PG4(TOSC1)
28	PG3(TOSC2)
70	PG2(ALE)
52	PG1(RD)
51	PG0(WR)
27	PH7(T4)
18	PH6(OC2B)
17	PH5(OC4C)
16	PH4(OC4B)
15	PH3(OC4A)
14	PH2(XCK2)
13	PH1(TXD2)
12	PH0(RXD2)
79	PJ7
69	PJ6(PCINT15)
68	PJ5(PCINT14)
67	PJ4(PCINT13)
66	PJ3(PCINT12)
65	PJ2(XCK3)
64	PJ1(TXD3)
63	PJ0(RXD3)
82	PK7(ADC15)
83	PK6(ADC14)
84	PK5(ADC13)
85	PK4(ADC12)
86	PK3(ADC11)
87	PK2(ADC10)
88	PK1(ADC9)
89	PK0(ADC8)
42	PL7
41	PL6
40	PL5(OC5C)
39	PL4(OC5B)
38	PL3(OC5A)
37	PL2(T5)
36	PL1(ICP5)
35	PL0(ICP4)
71	PA7(AD7)
72	PA6(AD6)
73	PA5(AD5)
74	PA4(AD4)
75	PA3(AD3)
76	PA2(AD2)
77	PA1(AD1)
78	PA0(AD0)
26	PB7(OC0A/OC1C)
25	PB6(OC1B)
24	PB5(OC1A)
23	PB4(OC2A)
22	PB3(MISO)
21	PB2(MOSI)
20	PB1(SCK)
19	PB0(SS)
60	PC7(A15)
59	PC6(A14)
58	PC5(A13)
57	PC4(A12)
56	PC3(A11)
55	PC2(A10)
54	PC1(A9)
53	PC0(A8)
50	PD7(T0)
49	PD6(T1)
48	PD5(XCK1)
47	PD4(ICP1)
46	PD3(TXD1/INT3)
45	PD2(RXD1/INT2)
44	PD1(SDA/INT1)
43	PD0(SCL/INT0)
9	PE7(ICP3/INT7)
8	PE6(T3/INT6)
7	PE5(OC3C/INT5)
6	PE4(OC3B/INT4)
5	PE3(OC3A/AIN1)
4	PE2(XCK0/AIN0)
3	PE1(TXD0)
2	PE0(RXD0)

# Digital Input/Output

- Each port has three special-purpose registers that control its behavior.
- For port B, they are:
  - DDRB: data direction register B
  - PORTB: port output register B
  - PINB: port input B

# Data Direction Register: DDRx

- 8-bit wide register
  - Controls one pin with each bit
- 0 -> this is an input pin
- 1 -> this is an output pin

# Port Output Register: PORTx

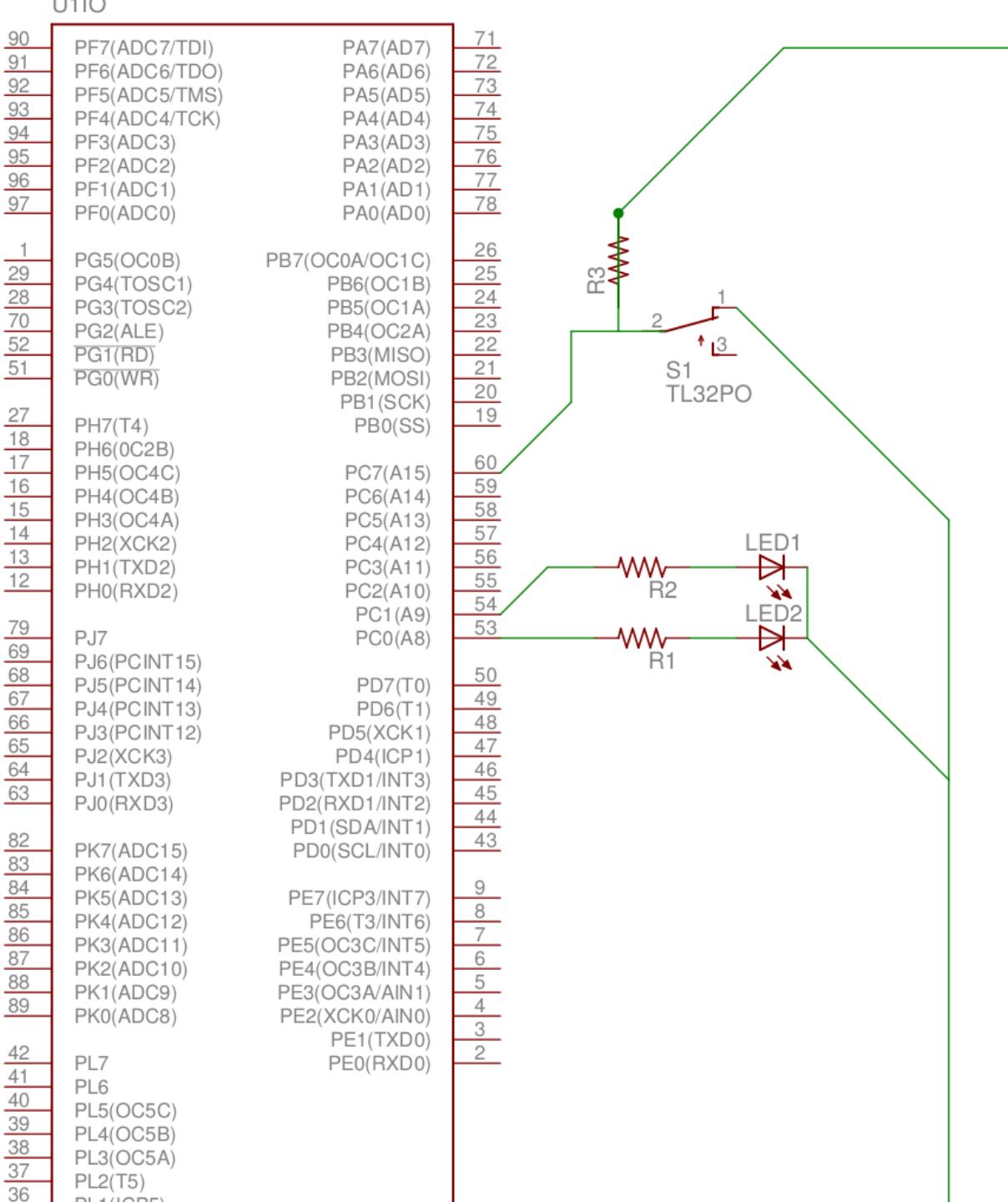
- Also one pin per bit
- If configured as an output:
  - 0 -> the pin is held at 0 V
  - 1 -> the pin is held at +5 V
- Note: only configure pins as an output if you really mean it!

# Port INput register: PINx

- One pin per bit
- Reading from the register:
  - 0 -> the voltage of the pin is near 0 V
  - 1 -> the voltage of the pin is near +5 V
- If nothing is connected to the pin, then the pin will appear to be in a random state

# A First Circuit

A

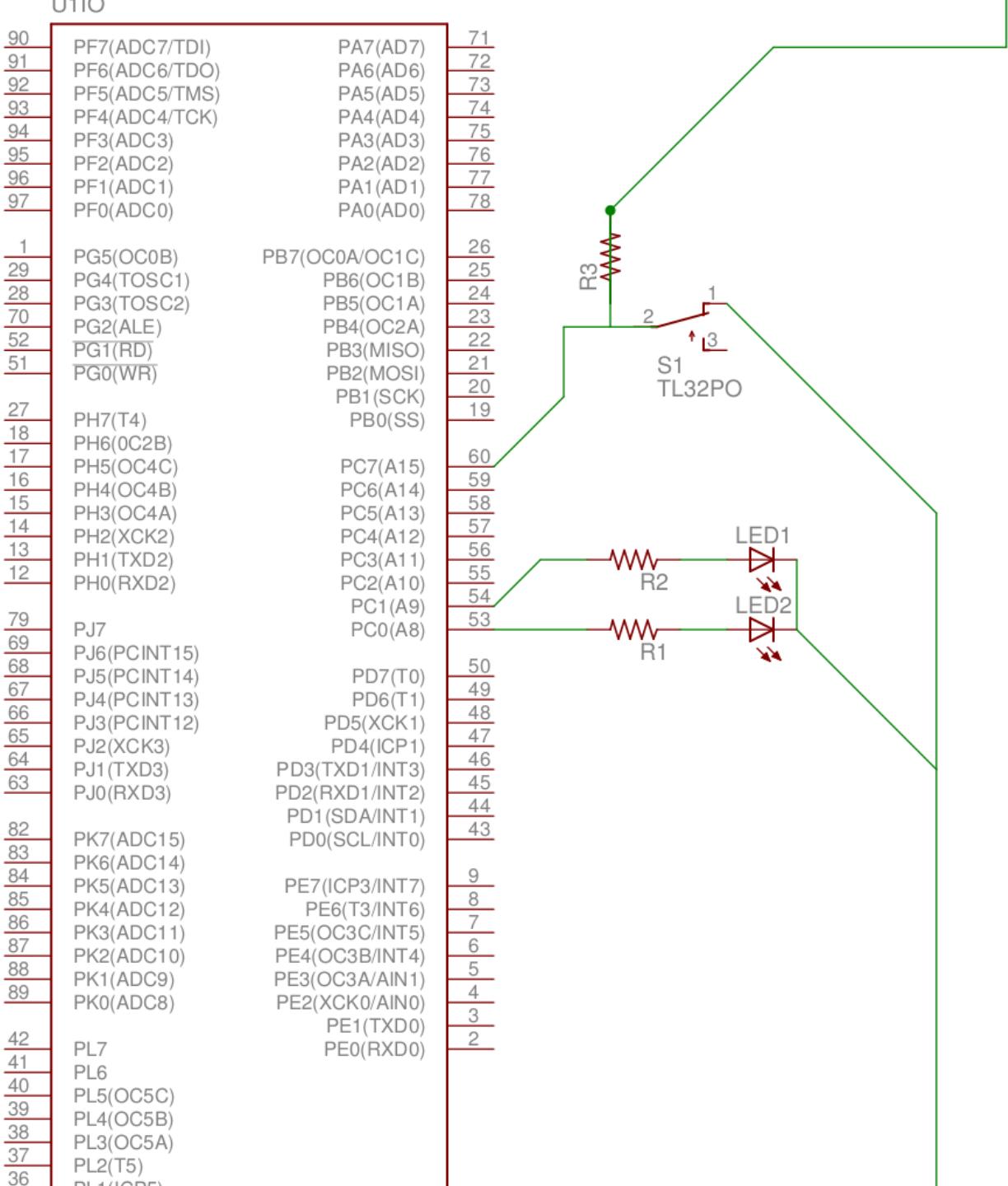


# A First Program

Flash the  
LEDs at a  
regular  
interval

- How do we  
do this?

A



# A First Program

```
main()  {
    DDRC = 0x3;

    while(1)
    {
        PORTC = 0x1;

        delay_ms(100);

        PORTC = 0x0;

        delay_ms(100);
    }

}
```

# A First Program

```
main() {  
    DDRC = 0x3;  
  
    while(1) {  
        PORTC = 0x1;           // sets PC0 to 1  
        delay_ms(100);  
        PORTC = 0x0;           // set PC0 to 0  
        delay_ms(100);  
    }  
}
```

# A First Program

```
main() {  
    DDRC = 1;    // Set port C pin 0 as an output  
  
    while(1) {  
        PORTC = PORTC ^ 0x1;    // XOR bit 0 with 1  
        delay_ms(500);        // Pause for 500 msec  
    }  
}
```

# A Second Program

```
main() {  
    DDRC = 3;    // Set port C pins 0, and 1 as outputs  
  
    while(1) {  
        PORTC = 0x3;  
        delay_ms(250);  
        PORTC = 0x1;  
        delay_ms(250);  
        PORTC = 0x2;  
        delay_ms(250);  
        PORTC = 0x0;  
        delay_ms(250);  
    }  
}
```

**What does this program do?**

# A Second Program

```
main() {  
    DDRC = 3;    // Set port C pins 0, and 1 as outputs  
  
    while(1) {  
        PORTC = 0x3;  
        delay_ms(250);  
        PORTC = 0x1;  
        delay_ms(250);  
        PORTC = 0x2;  
        delay_ms(250);  
        PORTC = 0x0;  
        delay_ms(250);  
    }  
}
```

**Flashes LED on PC1 at 2 Hz  
on PC0: 1 Hz**

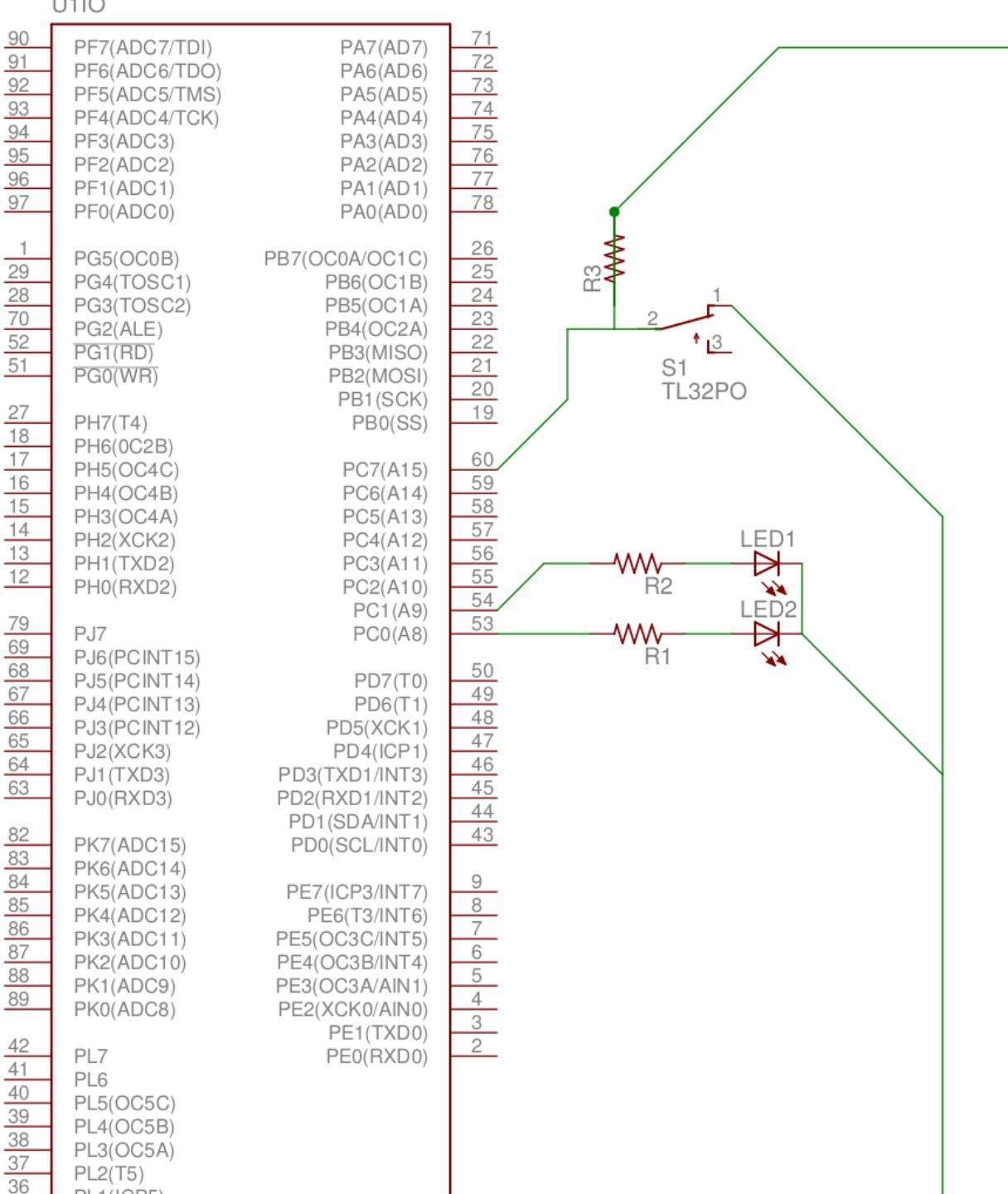
**Duty Cycle for each: 50%**

# A Third Program

If switch reads zero, turn PC0 on and PC1 off

Otherwise, turn PC0 off and PC1 on

A



# A Third Program

```
main() {  
    DDRC = 0x3;  
  
    while(1) {  
        if(PINC & 0x80) {  
            PORTC = 1;  
        } else {  
            PORTC = 2;  
        }  
    }  
}
```

# A Third Program

```
main()  {
    DDRC = 0x3;

    while(1)
    {
        if(PINC & 0x80) {
            PORTC = 0x2;
        }else{
            PORTC = 0x1;
        }
    }
}
```

# Port-Related Registers

Some of the C-accessible registers for controlling digital I/O:

	Directional control	Writing	Reading
Port B	DDRB	PORTB	PINB
Port C	DDRC	PORTC	PINC
Port D	DDRD	PORTD	PIND

# Arduino Mega Board

(see schematic)