1. **LED example**

| **DDRB** | **Description** |
| --- | --- |
| 0 | Input pin |
| 1 | Output pin |

| **PORTB** | **Description** |
| --- | --- |
| 0 | Output low value |
| 1 | Output high value |

| **DDRB** | **PORTB** | **Direction** | **Internal pull-up resistor** | **Description** |
| --- | --- | --- | --- | --- |
| 0 | 0 | Input | No | Tri-state, high-impedance |
| 0 | 1 | Input | Yes | PORTB will source current if ext. pulled low. |
| 1 | 0 | Output | No | Output low (Sink) |
| 1 | 1 | Output | No | Output high (Source) |

|  |
| --- |
|  |
| **Port** | **Pin** | **Input/output usage?** |
| A | x | Microcontroller ATmega328P does not contain port A |
| B | 0 | Yes (Arduino pin 8) |
|  | 1 | Yes (Arduino pin ~9) |
|  | 2 | Yes (Arduino pin ~10) |
|  | 3 | Yes (Arduino pin ~11) |
|  | 4 | Yes (Arduino pin 12) |
|  | 5 | Yes (Arduino pin 13) |
|  | 6 | No (xtal clock generator) |
|  | 7 | No (xtal clock generator) |
| C | 0 | Yes (Arduino pin A0) |
|  | 1 | Yes (Arduino pin A1) |
|  | 2 | Yes (Arduino pin A2) |
|  | 3 | Yes (Arduino pin A3) |
|  | 4 | Yes (Arduino pin A4) |
|  | 5 | Yes (Arduino pin A5) |
|  | 6 | No |
|  | 7 | No |
| D | 0 | Yes (Arduino pin RX<-0) |
|  | 1 | Yes (Arduino pin TX->0) |
|  | 2 | Yes (Arduino pin 2) |
|  | 3 | Yes (Arduino pin ~3) |
|  | 4 | Yes (Arduino pin 4) |
|  | 5 | Yes (Arduino pin ~5) |
|  | 6 | Yes (Arduino pin ~6) |
|  | 7 | Yes (Arduino pin 7) |

**LED example C code:**

1. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
2. \*
3. \* Alternately toggle two LEDs when a push button is pressed.
4. \* ATmega328P (Arduino Uno), 16 MHz, AVR 8-bit Toolchain 3.6.2
5. \*
6. \* Copyright (c) 2018-2020 Tomas Fryza
7. \* Dept. of Radio Electronics, Brno University of Technology, Czechia
8. \* This work is licensed under the terms of the MIT license.
9. \*
10. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/
11. /\* Defines -----------------------------------------------------------\*/
12. #define LED\_GREEN PB5 // AVR pin where green LED is connected
13. #define LED\_RED PC0
14. #define BTN PD0
15. #define BLINK\_DELAY 250
16. #ifndef *F\_CPU*
17. #define *F\_CPU* 16000000 // CPU frequency in Hz required for delay
18. #endif
19. /\* Includes ----------------------------------------------------------\*/
20. #include <util/delay.h> // Functions for busy-wait delay loops
21. #include <avr/io.h> // AVR device-specific IO definitions
22. /\* Functions ---------------------------------------------------------\*/
23. /\*\*
24. \* Main function where the program execution begins. Toggle two LEDs
25. \* when a push button is pressed.
26. \*/
27. int main(void)
28. {
29. /\* GREEN LED \*/
30. // Set pin as output in Data Direction Register...
31. DDRB = DDRB | (1<<LED\_GREEN);
32. // ...and turn LED off in Data Register
33. PORTB = PORTB & ~(1<<LED\_GREEN);
34. /\* second LED \*/
35. DDRC = DDRC | (1<<LED\_RED);
36. PORTC = PORTC & ~(1<<LED\_RED);
38. /\* button \*/
39. DDRD = DDRD & ~(1<<LED\_GREEN);
40. PORTD = PORTD | (1<<BTN);
42. // Infinite loop
43. while (1)
44. {
45. *\_delay\_ms*(BLINK\_DELAY);
47. loop\_until\_bit\_is\_clear(PIND, BTN);
49. PORTB = PORTB ^ (1<<LED\_GREEN);
50. PORTC = PORTC ^ (1<<LED\_RED);
51. }
52. // Will never reach this
53. return 0;
54. }
55. **KNIGHT RIDER application:**

**C code:**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* Alternately toggle two LEDs when a push button is pressed.

\* ATmega328P (Arduino Uno), 16 MHz, AVR 8-bit Toolchain 3.6.2

\*

\* Copyright (c) 2018-2020 Tomas Fryza

\* Dept. of Radio Electronics, Brno University of Technology, Czechia

\* This work is licensed under the terms of the MIT license.

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Defines -----------------------------------------------------------\*/

#define LED\_1 PB1

#define LED\_2 PB2

#define LED\_3 PB3

#define LED\_4 PB4

#define LED\_5 PB5

#define BTN PD0

#define BLINK\_DELAY 200

#ifndef *F\_CPU*

#define *F\_CPU* 16000000 // CPU frequency in Hz required for delay

#endif

/\* Includes ----------------------------------------------------------\*/

#include <util/delay.h> // Functions for busy-wait delay loops

#include <avr/io.h> // AVR device-specific IO definitions

/\* Functions ---------------------------------------------------------\*/

/\*\*

\* Main function where the program execution begins. Toggle two LEDs

\* when a push button is pressed.

\*/

int main(void)

{

/\* INITIALIZATION \*/

/\* LED 1 \*/

// Set pin as output in Data Direction Register...

DDRB = DDRB | (1<<LED\_1);

// ...and turn LED off in Data Register

PORTB = PORTB & ~(1<<LED\_1);

/\* LED 2 \*/

DDRB = DDRB | (1<<LED\_2);

PORTB = PORTB & ~(1<<LED\_2);

/\* LED 3 \*/

DDRB = DDRB | (1<<LED\_3);

PORTB = PORTB & ~(1<<LED\_3);

/\* LED 4 \*/

DDRB = DDRB | (1<<LED\_4);

PORTB = PORTB & ~(1<<LED\_4);

/\* LED 5 \*/

DDRB = DDRB | (1<<LED\_5);

PORTB = PORTB & ~(1<<LED\_5);

/\* button \*/

DDRD = DDRD & ~(1<<BTN);

PORTD = PORTD | (1<<BTN);

PORTB = PORTB ^ (1<<LED\_1); // turn LED1 on

*\_delay\_ms*(BLINK\_DELAY); // wait

// Infinite loop

while (1)

{

if (bit\_is\_clear(PIND, BTN))

{

/\* FORWARD \*/

PORTB = PORTB ^ (1<<LED\_1); // turn LED1 off

PORTB = PORTB ^ (1<<LED\_2); // turn LED2 on

*\_delay\_ms*(BLINK\_DELAY); // wait 200 ms

PORTB = PORTB ^ (1<<LED\_2); // turn LED2 off

PORTB = PORTB ^ (1<<LED\_3); // turn LED3 on

*\_delay\_ms*(BLINK\_DELAY); // wait 200 ms

PORTB = PORTB ^ (1<<LED\_3); // turn LED3 off

PORTB = PORTB ^ (1<<LED\_4); // turn LED4 on

*\_delay\_ms*(BLINK\_DELAY); // wait 200 ms

PORTB = PORTB ^ (1<<LED\_4); // turn LED4 off

PORTB = PORTB ^ (1<<LED\_5); // turn LED5 on

*\_delay\_ms*(BLINK\_DELAY); // wait 200 ms

/\* BACK \*/

PORTB = PORTB ^ (1<<LED\_5); // turn LED5 off

PORTB = PORTB ^ (1<<LED\_4); // turn LED4 on

*\_delay\_ms*(BLINK\_DELAY); // wait 200 ms

PORTB = PORTB ^ (1<<LED\_4); // turn LED4 off

PORTB = PORTB ^ (1<<LED\_3); // turn LED3 on

*\_delay\_ms*(BLINK\_DELAY); // wait 200 ms

PORTB = PORTB ^ (1<<LED\_3); // turn LED3 off

PORTB = PORTB ^ (1<<LED\_2); // turn LED2 on

*\_delay\_ms*(BLINK\_DELAY); // wait 200 ms

PORTB = PORTB ^ (1<<LED\_2); // turn LED2 off

PORTB = PORTB ^ (1<<LED\_1); // turn LED1 on

*\_delay\_ms*(BLINK\_DELAY); // wait 200 ms

}

}

// Will never reach this

return 0;

}