Přípravy na kroužek Mechatronika – KA6(A2d)

9. lekce – programování motorů, složitější pohyby

Časování délky chodu motoru

Jednoduché časování délky chodu motoru můžeme provádět pomocí funkce **_delay_ms(t)**, kde za **t** dosadíme požadovanou dobu v ms.

Příklad:

Založíme nový projekt v Atmel Studiu a v něm vytvoříme následující zdrojový kód:

```
#define F_CPU 8000000UL
#include <avr/io.h>
#include <util/delay.h>

int main(void)
{

	DDRD = ((1<<PD4)|(1<<PD5)|(1<<PD6)|(1<<PD7));

	while(1)
    {
		PORTD = ((1<<PORTD5)|(1<<PORTD6));
		_delay_ms(2000);
		PORTD &= ~((1<<PORTD5)|(1<<PORTD6));
		_delay_ms(2000);
	}
}
```

Program sestavíme, nahrajeme programátorem na základní desku a vyzkoušíme.

Princip funkce (popis programu):

Pomocí DDRD registru nastavíme příslušné kontakty MCU na výtup.

Na PORTD5 a PORTD6 nastavíme hodnotu 1, čím uvedeme vozítko do pohybu směrem vpřed (viz tabulky v lekci č. 7). Po uplynutí nastavené doby 2 s dojde k vynulování bitů na PORTD5 a PORTD6, vozítko se zastaví a bude čekat po nastavenou dobu (opět 2 s), pak se znovu rozjede.

S použitím funkce **_delay_ms(t)** lze zkombinovat pohyb vpřed a vzad. Program upravíme následovně:

Příklad:

```
#define F_CPU 8000000UL
#include <avr/io.h>
#include <util/delay.h>

int main(void)
{

	DDRD = ((1<<PD4)|(1<<PD5)|(1<<PD6)|(1<<PD7));

	while(1)
    {
		PORTD = ((1<<PORTD5)|(1<<PORTD6));
		_delay_ms(2000);
		PORTD &= ~((1<<PORTD5)|(1<<PORTD6));
		PORTD = ((1<<PORTD4)|(1<<PORTD7));
		_delay_ms(2000);
		PORTD &= ~((1<<PORTD4)|(1<<PORTD7));
	}
}
```

Princip funkce (popis programu):

Princip je stejný jako u předchozího programu, pouze místo čekání se vozítko pohybuje vzad, tj. mělo by se po nastavené době vrátit vždy zpět do výchozí pozíce na startu.

Obdobně lze vyřešit zatáčení vozítka:

Příklad:

```
#define F_CPU 8000000UL
#include <avr/io.h>
#include <util/delay.h>

int main(void)
{

	DDRD = ((1<<PD4)|(1<<PD5)|(1<<PD6)|(1<<PD7));

	while(1)
    {
		PORTD = ((1<<PORTD5)|(1<<PORTD6));
		_delay_ms(2000);
		PORTD ^= (1<<PORTD6);
		_delay_ms(500);
		PORTD ^= (1<<PORTD6);
	}
}
```

Princip funkce (popis programu):

Po nastavené době jízdy vpřed provedeme vynulování bitu na PORTD6, tím zastavíme jeden z motorů. Vzhledem k tomu, že druhý z motorů zůstává stále v chodu, začne vozítko zatáčet, v ideálním případě se podle našeho příkladu otočí o 180° a dojede zpět na výchozí pozici.

Po přidání cyklu **for** bude zatáčení pomalejší (plynulejší):

```
#define F_CPU 8000000UL
#include <avr/io.h>
#include <util/delay.h>

int main(void)
{

	DDRD = ((1<<PD4)|(1<<PD5)|(1<<PD6)|(1<<PD7));

	while(1)
    {
		PORTD = ((1<<PORTD5)|(1<<PORTD6));
		_delay_ms(20);
		for (uint8_t i=0;i<=10;i++) {
			PORTD ^= (1<<PORTD6);
			_delay_ms(10);
		}
	}
}
```

Princip funkce (popis programu):

V cyklu **for** se vždy na malou chvíli jeden z motorů zastaví (přepnutí bitu PORTD6) a poté na chvíli rozjede. Tím je dosaženo plynulejšího zatáčení a většího poloměru zatáčky.

Závěr:

Kombinací všech těchto způsobů se může vozitko pohybovat po libovolných drahách.

Vypracoval Radek Zvěřina. Použité materiály: Merkur.