|  |  |  |  |
| --- | --- | --- | --- |
| Systemy wbudowane Laboratorium | | | |
| **Grupa:**  B | **Temat:**  Instrukcje sterujące | | |
| **Data:**  08.04.2021r. | **Wykonał:**  Piotr Drabik | | |
| **Godzina:**  12:40 | **II rok**  **Informatyk Stosowana** | **Ocena i uwagi prowadzącego:** | **Prowadzący:**  dr hab. Witold Kozłowski |

### Opis zadania:

Celem zadania jest zbadanie zależności czasowych instrukcji ustawiających stan portów na mikrokontrolerze. Określimy również ilość cykli pracy mikrokontrolera. Wartości odczytywać będziemy z symulatora zawartego w programie BASCOM-AVR oraz z oscyloskopu.

Podczas zadania sprawdzimy wartości czasowe równoważnego programu napisanego w języku ASSEMBLER. W tym wypadku wartości odczytamy jedynie z oscyloskopu.

Porty w mikrokontrolerze służą do jego komunikacji z otoczeniem i co ważne, porty te są dwukierunkowe, więc mogą być albo wejściami albo wyjściami.

### Instrukcje Set/Reset:

Aby sprawdzić działanie instrukcji **Set** i **Reset** musimy przygotować kod:

**$regfile** = "m8def.dat" Informuje kompilator o pliku dyrektyw mikroprocesora

**$crystal**  = 8000000 Informuje kompilator o częstotliwości oscylatora taktującego mikrokontroler

**Config** PINB.0 = **Output** Ustawiamy Pin B 0 jako wyjście

**Do** Początek pętli

**Set** Portb.0 Ustawiamy wyjście PB0 na „**1**”

**Reset** PORTB.0 Ustawiamy wyjście PB0 na „**0**”

**Loop** Koniec pętli głównej programu

**End** Koniec programu

­

#### Zależności czasowe instrukcji Set/Reset odczytane z symulatora programu BASCOM-AVR

|  |  |  |
| --- | --- | --- |
| **Instrukcje** | **Czas [mS]** | **Cykle** |
| Set | 0.77025 | 6162 |
| Reset | 0.7705 | 6164 |
| Loop | 0.77075 | 6166 |
| Set | 0.771 | 6168 |
| Reset | 0.77125 | 6170 |
| Loop | 0.7715 | 6172 |

Jak wynika z powyższej tabeli, zarówno instrukcja Set, jak i Reset wykonywane są w dwóch taktach zegara. Widać również, że każda z tych instrukcji zajmuje **0,00025mS**, czyli **250nS**.

Podłączając oscyloskop do naszego układu możemy odczytać, że stan „**1**” na oscyloskopie trwa **252nS**, a zbocze opadające( „**0**” ) – **500nS,** wartość ta jednak wynika z wykonania instrukcji Reset w dwóch taktach, oraz instrukcji Loop również w dwóch taktach**.** Wynika z tego czas wykonywania instrukcji **Reset = 250nS** oraz **Loop = 250nS.** Wartości te pokrywają się prawie idealnie z wartościami odczytanymi z oscyloskopu.

Wynika z tego, że okres zapalenia i zgaszenia diody trwa **T =** **252nS + 500nS = 752nS**, więc możemy policzyć częstotliwość:

f = = = 1,3297MHz,

z czego wynika, że w ciągu jednej sekundy dioda zapali się i zgaśnie **1 329 700** razy.

### Instrukcja Toggle

Aby sprawdzić działanie instrukcji **Toggle** musimy przygotować kod:

**$regfile** = "m8def.dat" Informuje kompilator o pliku dyrektyw mikroprocesora

**$crystal**  = 8000000 Informuje kompilator o częstotliwości oscylatora taktującego mikrokontroler

**Config** PINB.0 = **Output** Ustawiamy PB0 jako wyjście

**Do** Początek pętli

**Toggle** PORTB.0 Zmiana na stan przeciwny linii PB0

**Loop** Koniec pętli głównej programu

**End** Koniec programu

#### Zależności czasowe instrukcji Toggle odczytane z symulatora programu BASCOM-AVR

|  |  |  |
| --- | --- | --- |
| **Instrukcje** | **Czas [mS]** | **Cykle** |
| Loop | 0.77125 | 6170 |
| Toggle | 0.7715 | 6172 |
| Loop | 0.772 | 6176 |
| Toggle | 0.77225 | 6178 |

Z powyższej tabeli można odczytać, że instrukcja **Toggle** wykonuje się w czterech taktach zegara, oraz że czas wykonania instrukcji **Toggle** wynosi **0,0005mS**, czyli **500nS**.

Po podłączeniu oscyloskopu do naszego układu możemy odczytać, że stan „**1**” trwa **744nS**, podobnie jak czas „**0**” logicznego – również **744nS.**

Wynika z tego, że okres zapalenia i zgaszenia diody trwa **T =** **744nS + 744nS = 1 488nS**, więc możemy policzyć częstotliwość:

f = = = 0,62720MHz,

z czego wynika, że w ciągu jednej sekundy dioda zapali się i zgaśnie **627 200** razy.

Widzimy, że Toggle miga ponad dwa razy wolniej niż instrukcje Set/Reset.

### Program w języku ASSEMBLER

Aby sprawdzić działanie programu w języku ASSEMBLERmusimy przygotować kod:

.nolist

.include”m8def.inc”

**ldi** r16, 0b00000001

**out** ddrb, r16

**ldi** r17, 0b00000000

loop:

**out** portb,r16

**out** portb,r17

**rjmp** loop

.exit

Wartości dla programu napisanego w ASSEMBLERZE możemy odczytać jedynie za pomocą oscyloskopu. Obserwujemy, że czas trwania „**1**” logicznej wynosi **128nS**, a czas trwania „**0**” logicznego wynosi **360nS**.

Wynika z tego, że okres zapalenia i zgaszenia diody trwa **T =** **128nS + 360nS = 488nS**, więc możemy policzyć częstotliwość:

f = = = 2,04918MHz,

z czego wynika, że w ciągu jednej sekundy dioda zapali się i zgaśnie **2 049 180** razy.

### Podsumowanie

Jak możemy zauważyć, program napisany w języku ASSEMBLER ma najwyższą częstotliwość z badanych programów, a najwolniejszy jest program napisany z użyciem instrukcji Toggle.