

Temat 6. Pakiet MPLAB IDE

Pakiet MPLAB IDE jest bezpłatnym zintegrowanym środowiskiem programistycznym dla Mikrokontrolerów z rodziny Microchip PIC, który umożliwia:

- pisanie programów,
- kompilowanie programów w językach C i assembler,
- sterowanie programatorami mikrokontrolerów,
- symulację programową mikrokontrolera (bez dodatkowych układów zewnętrznych).

Skąd można pobrać instalator MPLAB IDE?

1). Archiwum programów na stronie producenta:

<https://www.microchip.com/development-tools/pic-and-dspic-downloads-archive>

(nie wymaga rejestracji)

2). Materiały do przedmiotu Technika cyfrowa:

<https://fizyka.p.lodz.pl/pl/dla-studentow/fizyka-tech/tc/>

6.1. Instalacja MPLAB IDE

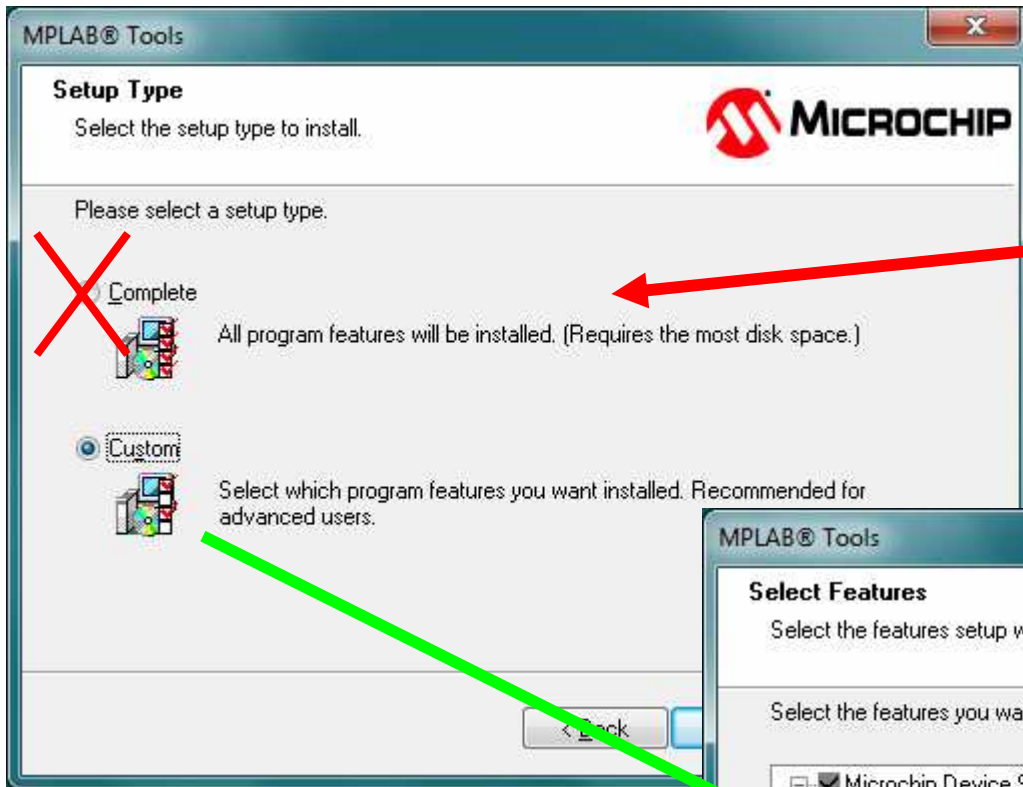
Zalecana jest starsza wersja MPLAB IDE 8.66 + HI-TECH C Compiler 9.81, bo:

- nowsze wersje nie zawierają załączonego kompilatora HI-TECH C,
- mogą mieć odmienną obsługę niż opisano w materiałach do zajęć,
- najnowszy pakiet MPLAB IDE X nie obsługuje starszych programatorów (w tym Picstart PLUS na wyposażeniu laboratorium techniki cyfrowej).



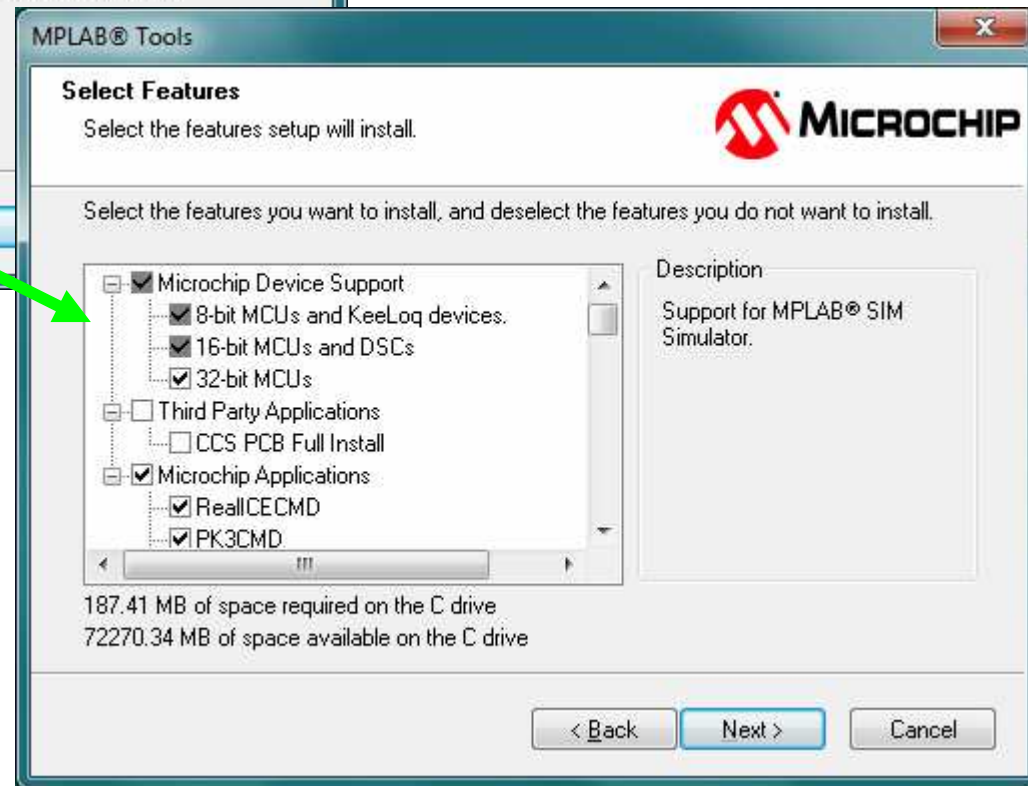
Rys. 6.1. Okno startowe instalatora.

Instalacja MPLAB IDE c.d. ...



Instalacja pełna - nie zalecana
instaluje mnóstwo składników
zbędnych na zajęciach
z Techniki Cyfrowej

Po wyborze „Custom”
wiele zbędnych składników
jest domyślnie wybranych.



Instalacja MPLAB IDE c.d. ...

- składniki niezbędne do zajęć Techniki Cyfrowej

- Microchip Device Support
 - 8-bit MCUs and KeeLoq devices.
 - 16-bit MCUs and DSCs
 - 32-bit MCUs
- Third Party Applications
 - CCS PCB Full Install
- Microchip Applications
 - ReallCECMD
 - PK3CMD
 - ICD3CMD
 - Procmd
 - Visual Procmd
 - PM3Cmd
 - MPASM Suite
 - ASM30 Suite
 - MPLAB C32 Suite
 - HI-TECH C for PIC10/12/16
 - Application Maestro
- MPLAB IDE
 - AN851 - Bootloader
 - MPLAB PM3
 - PICSTART Plus
 - PRO MATE II
 - MPLAB ICD 2
 - MPLAB ICD 3
 - MPLAB SIM
 - PICKit
 - PICKit 2
 - PICKit 3 Programmer/Debugger
 - MPLAB ICE 2000
 - MPLAB ICE 4000
 - MPLAB REAL ICE
 - KEELOQ
 - PIC32MX Starter Kit
 - MPLAB Starter Kit for Serial Memory Products
 - Starter Kit Support
- MPLAB IDE Tools

8-bit MCUs - urządzenia 8-bitowe

CCS PCB - opcjonalnie?

Ten kompilator nie oferuje pełnego języka C

MPASM Suite

assembler urządzeń 8-bitowych

HI-TECH C for PIC10/12/16

zalecany kompilator języka C

PICSTART Plus

programator mikrokontrolerów
dostępny w pracowni

MPLAB SIM

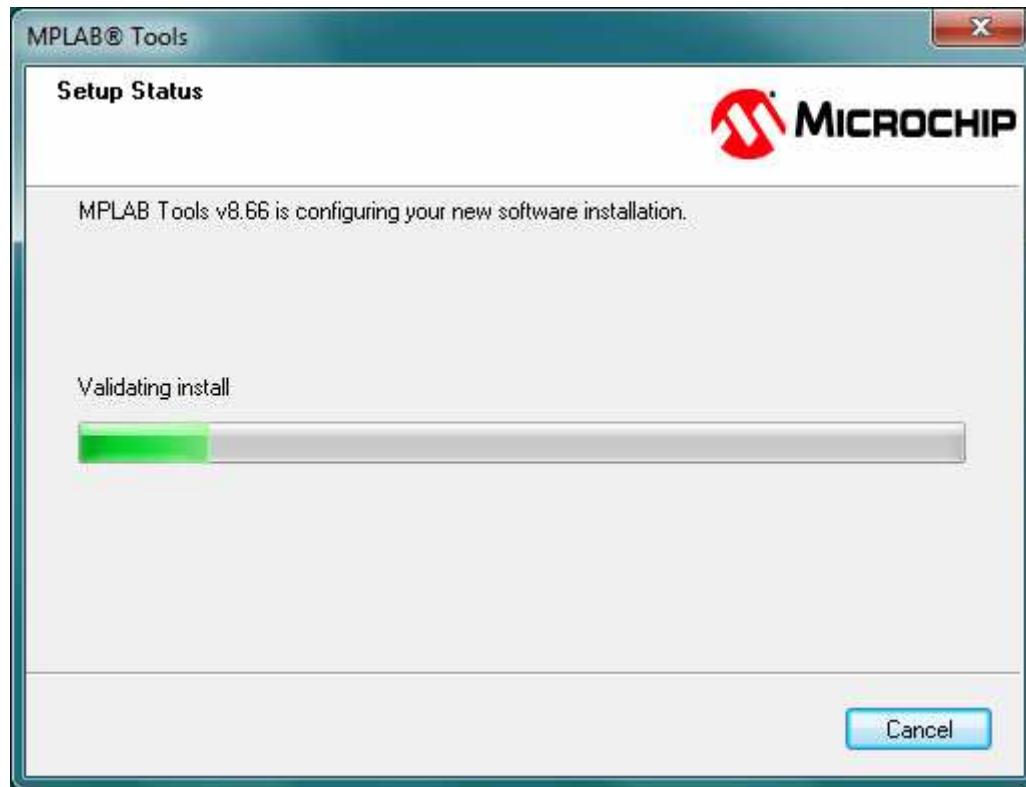
symulator
programowy
mikrokontrolera

Przydatne narzędzia kontroli
i wyświetlania

- MPLAB IDE Tools
 - MPLAB Target Application In/Out Display
 - Data Monitor and Control Interface
 - AN908 ACIM Tuning Interface
 - RTOS Viewer
 - MATLAB
 - Gimpel PC-Lint/MISRA Plug-in
 - SMPS GUIs
 - LCD Designer
 - dsPIC Filter Designer
 - mTouch
 - dsPIC Works Plug-in

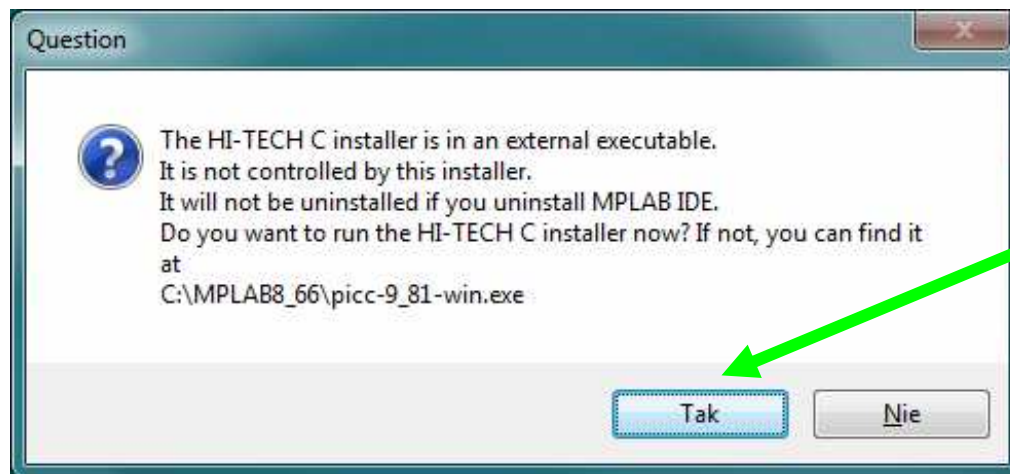
Instalacja MPLAB IDE c.d. ...

1)



Instalator pakietu MPLAB wywołuje zewnętrzny instalator dla kompilatora HI-TECH C

2)



Wybrać „Tak” żeby zainstalować kompilator HI-TECH C

6.2. Szybkie rozpoczęcie pracy w MPLAB IDE

Krok 1

Decyzja czy tworzymy projekt?

program w standardowym
aseblerze MPASM

program w innym języku, np.:
- HI-TECH C for PIC10/12/16,
- CCS C Compiler

program jednoplikowy
można skompilować
bez projektu

możliwe utworzenie
projektu

konieczne utworzenie
projektu

✓ **zalecane w instrukcji
do ćw. E58**

Kompilacja bez projektu odbywa się
w kontekście bieżących ustawień
w MPLAB IDE.

✓ **instrukcja do ćw. E59**

W pliku projektu *.mcp zapisane są m.in.:

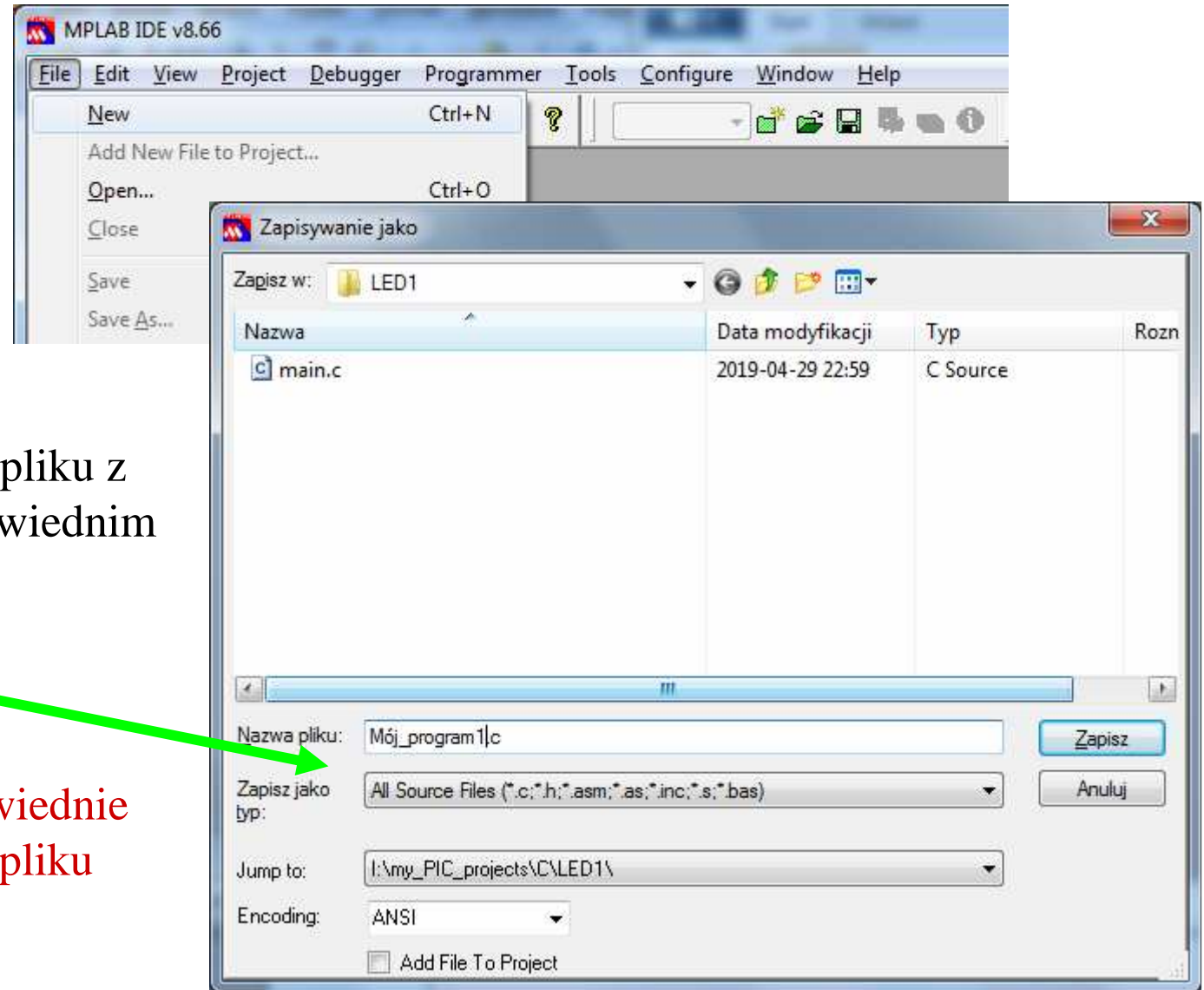
- lista plików źródłowych programu,
- wybór mikrokontrolera i bity konfiguracyjne,
- opcje kompilatora.

Szybkie rozpoczęcie pracy w MPLAB IDE c.d. ...

wersja uproszczona tworzenia bez projektu

Krok 2

Edycja nowego programu albo otwarcie istniejącego pliku



Krok 3

Program zapisać do pliku z rozszerzeniem odpowiednim dla języka, np.:

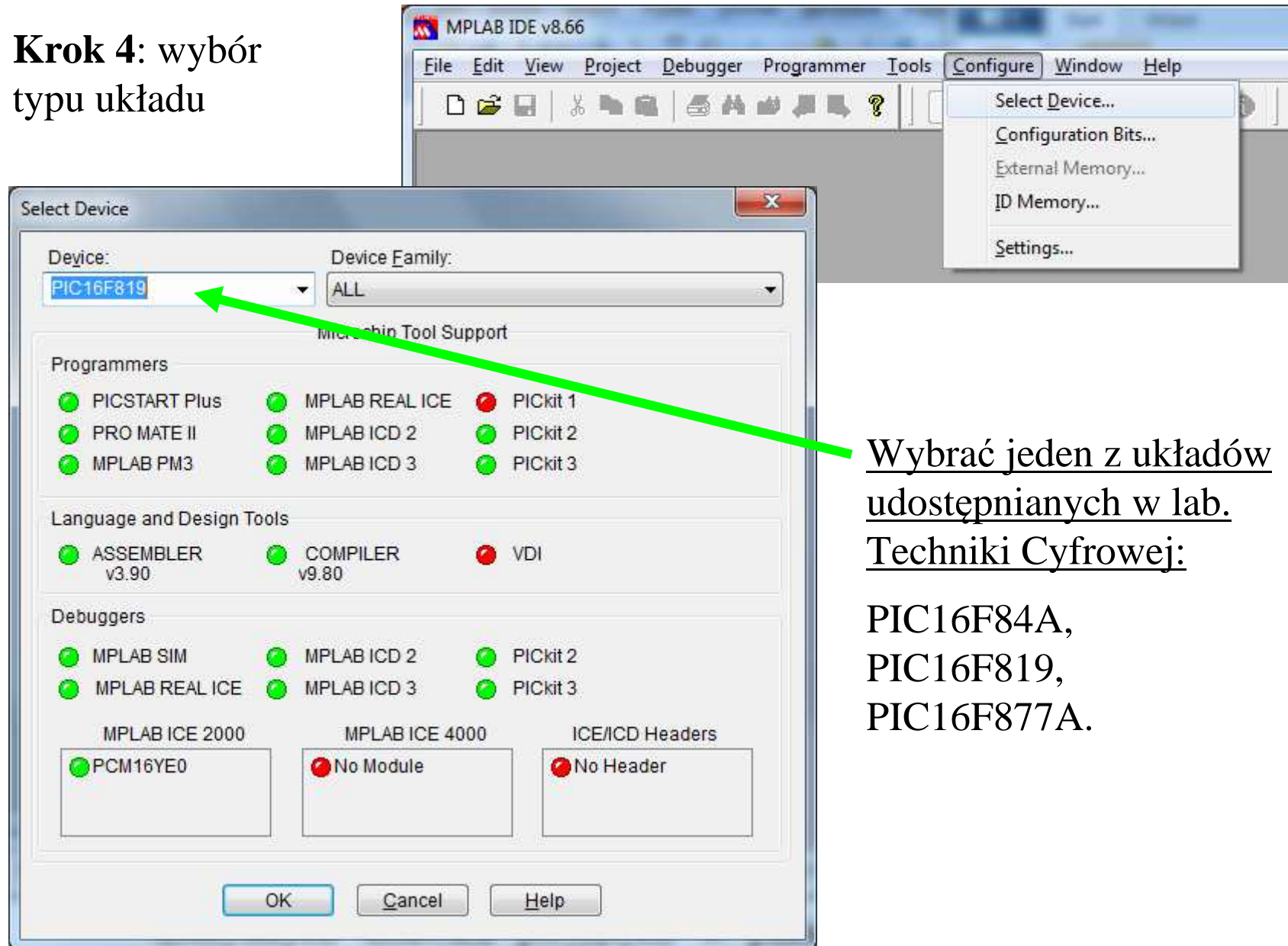
.asm – asembler

UWAGA: nieodpowiednie rozszerzenie nazwy pliku uniemożliwi użycie kompilatora.

Szybkie rozpoczęcie pracy w MPLAB IDE c.d. ...

wersja uproszczona tworzenia bez projektu

Krok 4: wybór typu układu

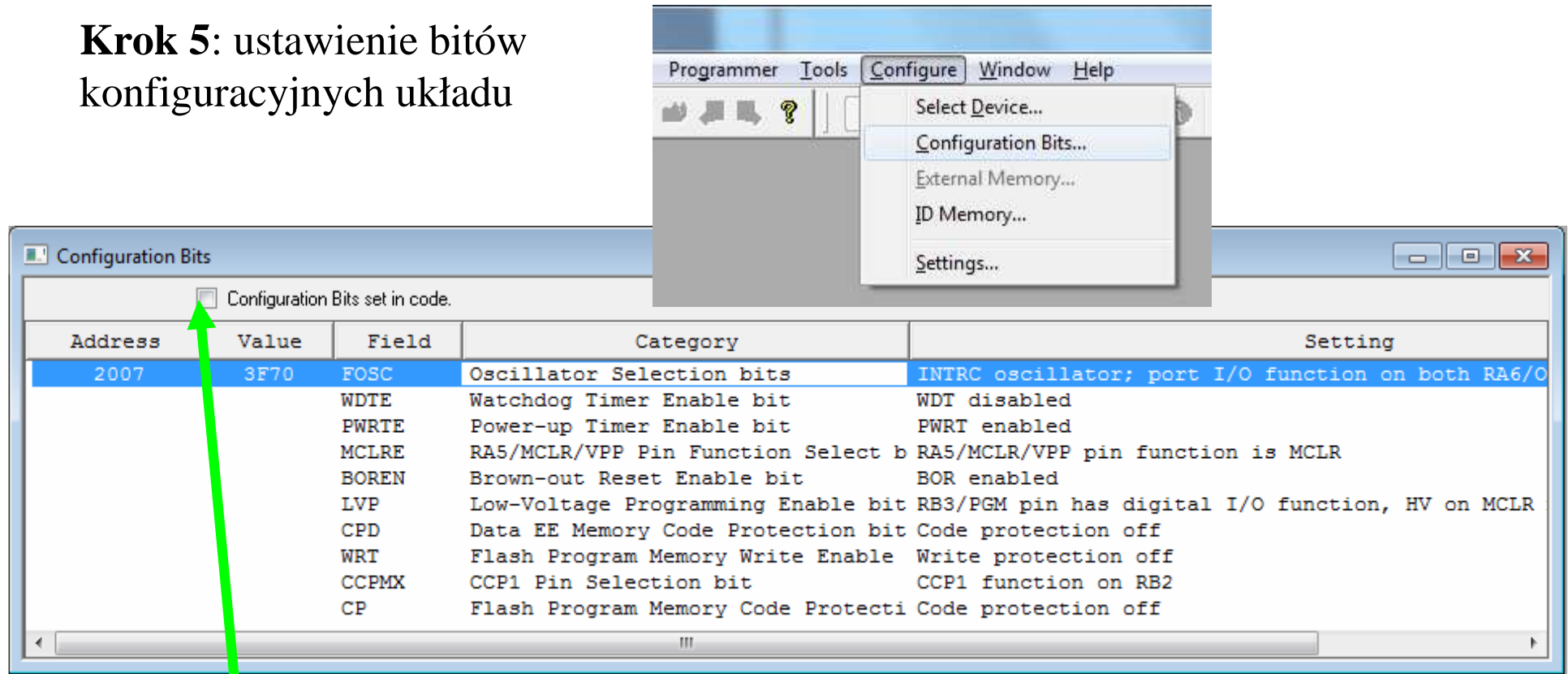


Wybrać jeden z układów udostępnianych w lab. Techniki Cyfrowej:

PIC16F84A,
PIC16F819,
PIC16F877A.

Szybkie rozpoczęcie pracy w MPLAB IDE c.d. ...

Krok 5: ustawienie bitów konfiguracyjnych układu



Jeśli włączone, to bity konfiguracyjne są ustawiane komendą `__config` umieszczoną w tekście programu.

UWAGA: pozostawienie bez zmian początkowych ustawień bitów konfiguracyjnych może okazać się błędnym wyborem, który uniemożliwi działanie programu!

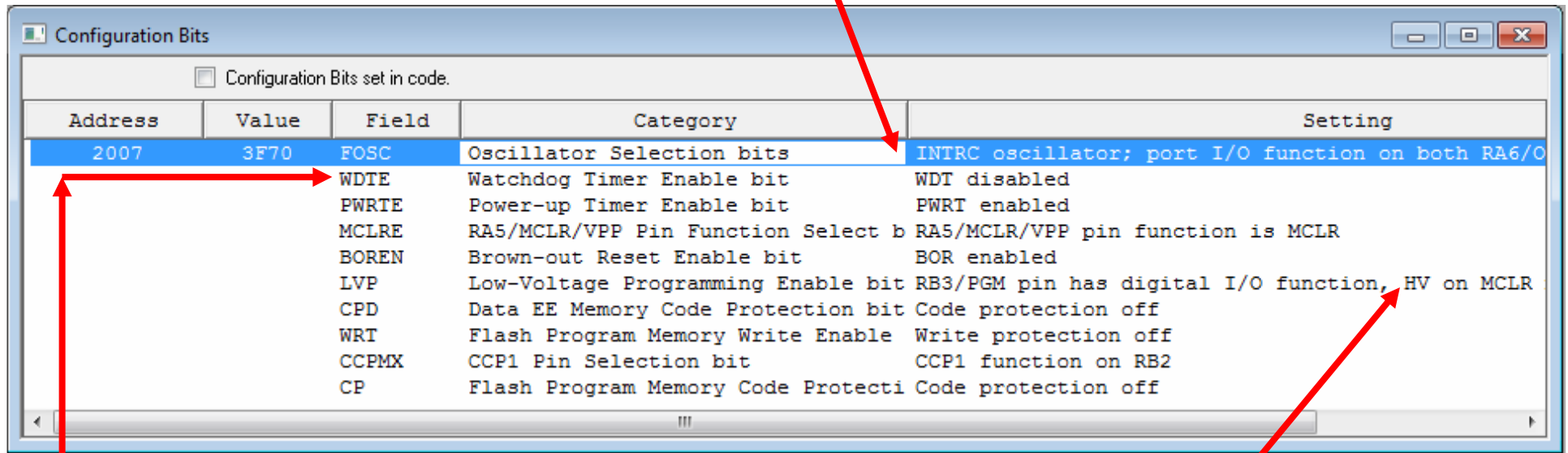
Należy zapoznać się ze znaczeniem poszczególnych bitów konfiguracyjnych opisanym w instrukcji do ćw. E58, rozdział 3.3.

Szybkie rozpoczęcie pracy w MPLAB IDE c.d. ...

Krok 5 c.d. ...

szczególną uwagę zwrócić na:

Zestaw uruchomieniowy ZL4PIC umożliwia tylko:
XT – oscylator kwarcowy średniej częstotliwości,
INTRC – wewnętrzny oscylator RC.



Wyłączyć Watchdog Timer, tzn. wybrać „WDT disabled”.

Watchdog Timer jest zegarem, który po przepełnieniu licznika powoduje reset mikrokontrolera. Poprawne używanie Watchdog Timer wymaga jawnego zerowania licznika wewnątrz każdej długo trwającej pętli programu.

Opis wszystkich bitów konfiguracyjnych znajduje się w instrukcji do ćwiczenia laboratoryjnego E58 [4].

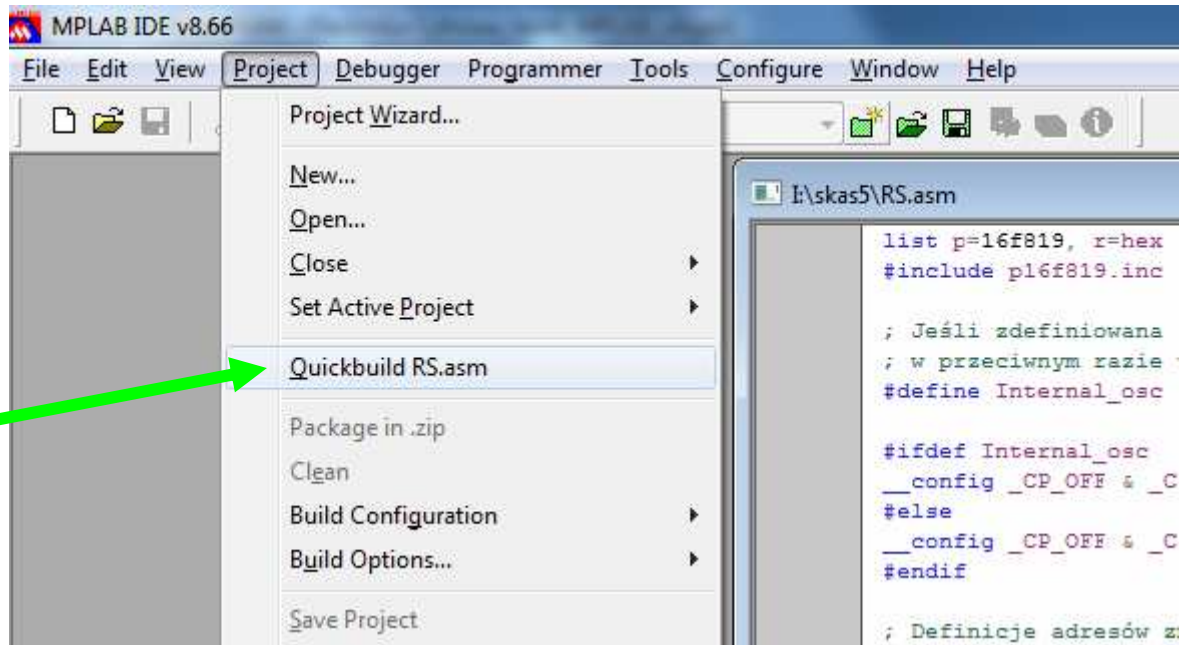
Wybrać HV (High-voltage programming), w przeciwnym razie pin RB3 mikrokontrolera nie działała jako wejście/wyjście podczas wykonywania programu.

Szybkie rozpoczęcie pracy w MPLAB IDE c.d. ...

wersja uproszczona tworzenia bez projektu

Krok 6. Kompilacja programu w asemblerze

Quickbuild kompiluje program z aktywnego okna edytora

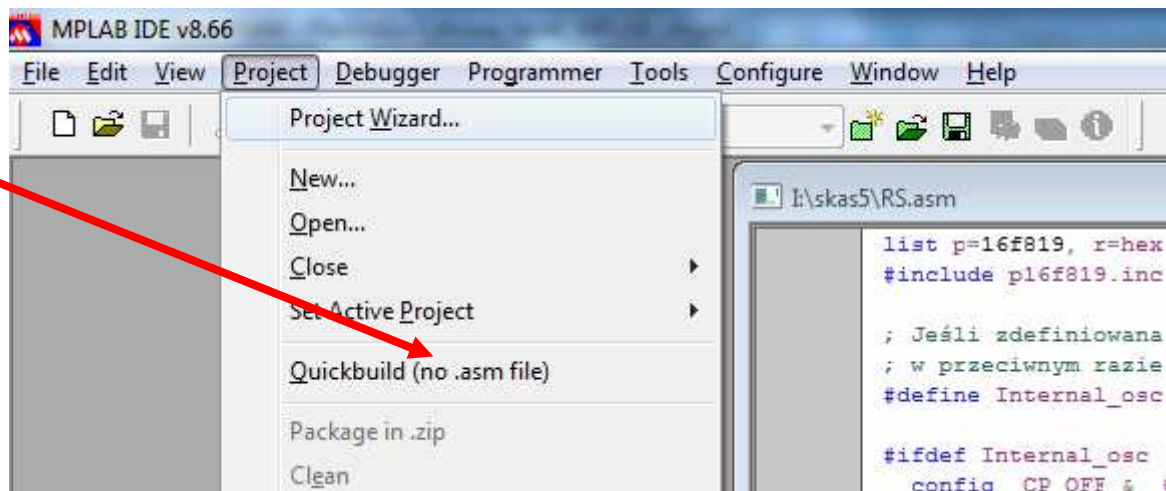


Co się stało?

Nie można skompilować programu otwartego w edytorze!

Rozwiązanie

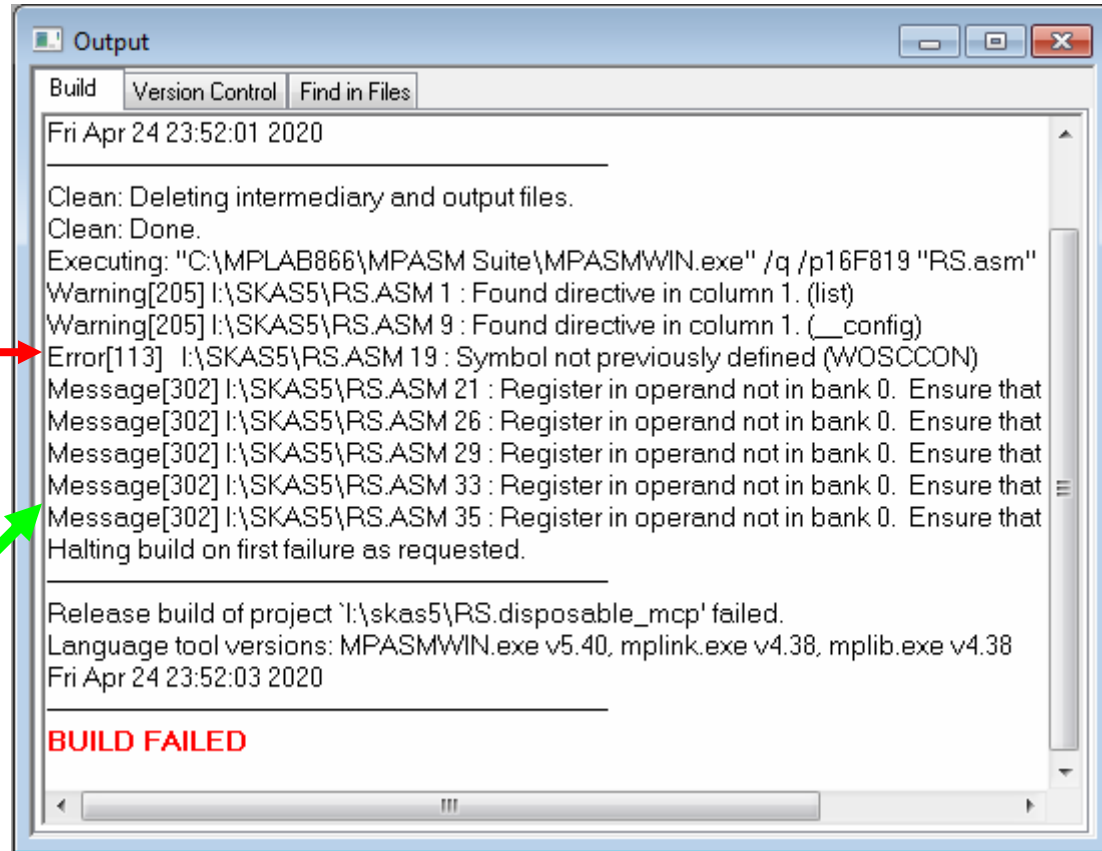
Okno z programem musi być wybrane (można w nim pisać z klawiatury, widoczny kursor).



Szybkie rozpoczęcie pracy w MPLAB IDE c.d. ...

Krok 7. Poprawianie błędów w programie.

Dwukliknięcie przenosi szybko do miejsca błędu w edytorze



```
Output
Build Version Control Find in Files
Fri Apr 24 23:52:01 2020
Clean: Deleting intermediary and output files.
Clean: Done.
Executing: "C:\MPLAB866\MPASM Suite\MPASMWIN.exe" /q /p16F819 "RS.asm"
Warning[205] I:\SKAS5\RS.ASM 1 : Found directive in column 1. (list)
Warning[205] I:\SKAS5\RS.ASM 9 : Found directive in column 1. (__config)
Error[113] I:\SKAS5\RS.ASM 19 : Symbol not previously defined (WOSCCON)
Message[302] I:\SKAS5\RS.ASM 21 : Register in operand not in bank 0. Ensure that
Message[302] I:\SKAS5\RS.ASM 26 : Register in operand not in bank 0. Ensure that
Message[302] I:\SKAS5\RS.ASM 29 : Register in operand not in bank 0. Ensure that
Message[302] I:\SKAS5\RS.ASM 33 : Register in operand not in bank 0. Ensure that
Message[302] I:\SKAS5\RS.ASM 35 : Register in operand not in bank 0. Ensure that
Halting build on first failure as requested.
Release build of project 'I:\skas5\RS.disposable_mcp' failed.
Language tool versions: MPASMWIN.exe v5.40, mplink.exe v4.38, mplib.exe v4.38
Fri Apr 24 23:52:03 2020
BUILD FAILED
```

Liczne ostrzeżenia (*ang. warning*) i wiadomości (*ang. message*) nie muszą oznaczać błędów koniecznych do poprawienia.

Szybkie rozpoczęcie pracy w MPLAB IDE c.d. ...

Krok 8.

Co zrobić ze skompilowanym programem?

1). Uruchomić program w symulatorze MPLAB SIM (rozdział 6.3).

2). Zapisać program do pamięci mikrokontrolera (dostępne na zajęciach w pracowni).

3). Kompilator zapisuje w folderze programu plik wyjściowy *.hex.

Ten plik można zaimportować do MPLAB żeby:

- przejrzeć kod programu (dostępny odwrotny assembler),
- przejrzeć ustawienia bitów konfiguracyjnych mikrokontrolera,
- uruchomić program w symulatorze,
- zapisać do pamięci mikrokontrolera.

6.3. Programowa symulacja mikrokontrolera

Co to jest MPLAB SIM ?

MPLAB SIM jest symulatorem mikrokontrolerów PIC i dsPIC razem z większością ich wbudowanych układów peryferyjnych. Umożliwia:

- + wykonywanie ciągle programu oraz śledzenie krokowe,
- + ustawianie wartości rejestrów lub ich pojedynczych bitów,
- + śledzenie bieżących wartości rejestrów, pamięci i stosu wywołań.

MPLAB SIM nie umożliwia:

- symulacji zewnętrznych układów wej./wyj., np. wyświetlacze numeryczne i alfanumeryczne, głośnik, realne przyciski z wibracją styków...
- symulacji wej./wyj. na poziomie pinów układu, dostępny jest tylko poziom rejestrów, np. RB0 reprezentuje wartość w bicie 0 rejestru PORTB, a nie wartość odczytaną z pinu o nazwie RB0 (w realnych układach podłączone pojemności i zwarcia powodują różnicę pomiędzy wartością ustawioną na pinie i po chwili odczytaną),
- funkcjonalność przetwornika A/D i komparatorów analogowych jest symulowana na poziomie rejestrów, co jednak trudno wykorzystać z powodu braku możliwości zadawania dowolnych napięć na wejściach analogowych.

Programowa symulacja mikrokontrolera

Krok 1. Otwarcie programu do wykonania

Wariant A – ze źródła

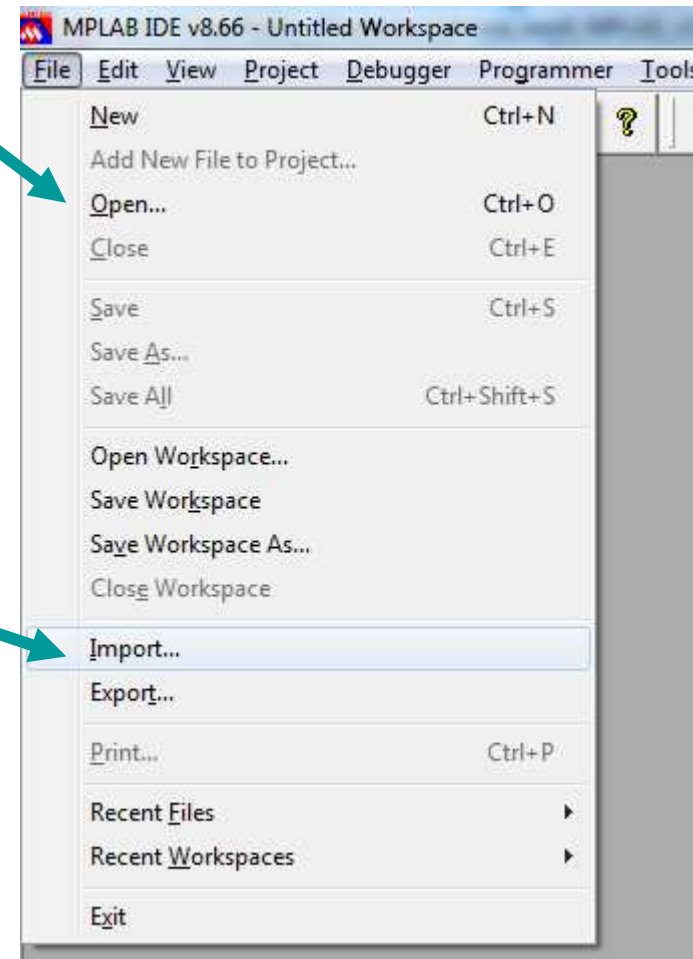
- otworzyć źródło programu,
- ustawić typ układu,
- ustawić bity konfiguracyjne,
- skompilować program

Postępowanie opisane wcześniej rozdziale 6.2

Wariant B – z pliku z kodem

- ustawić najpierw typ układu,
- zaimportować kod z pliku *.hex

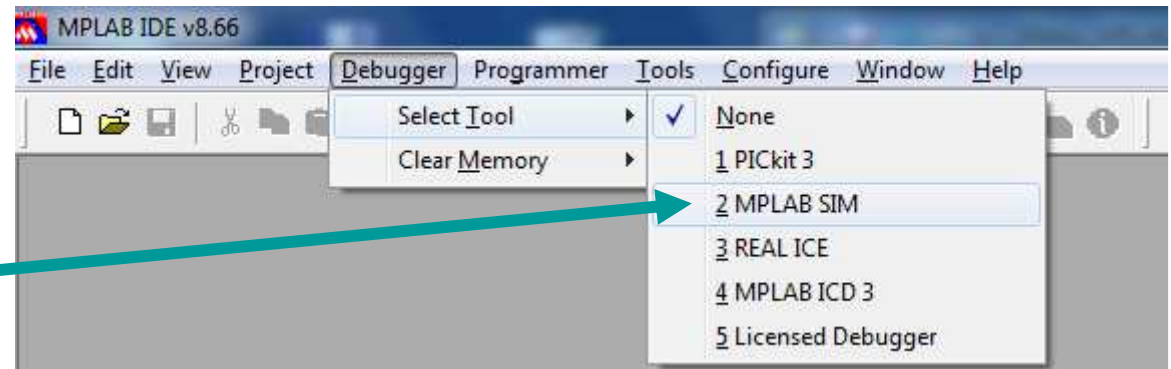
Bity konfiguracyjne są wczytywane z pliku *.hex



Programowa symulacja mikrokontrolera

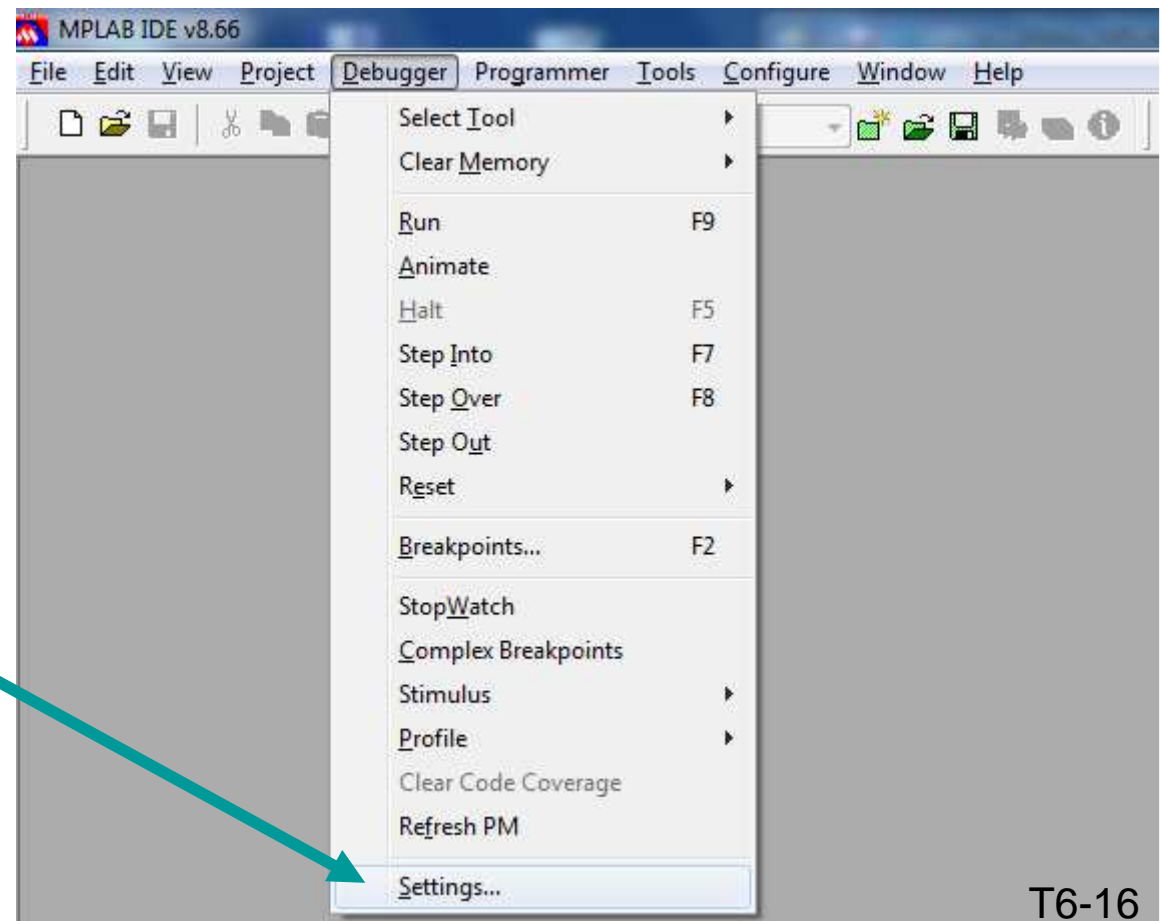
Krok 2. Wybór symulatora

- Wybrać MPLAB SIM jako aktywne narzędzie.



Menu Debugger dostosuje się samoczynnie do wybranego narzędzia

- Otworzyć menu Settings... (ustawienia)



Programowa symulacja mikrokontrolera

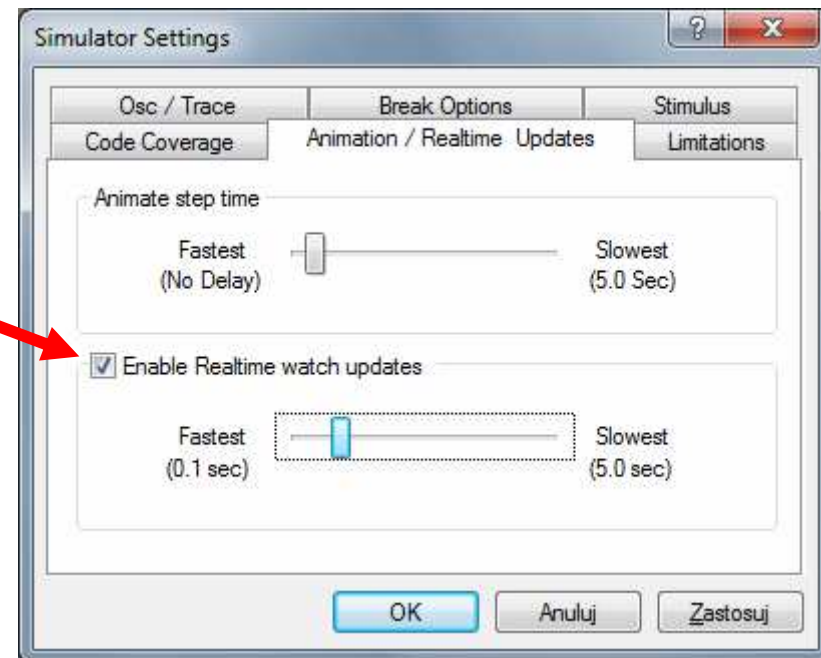
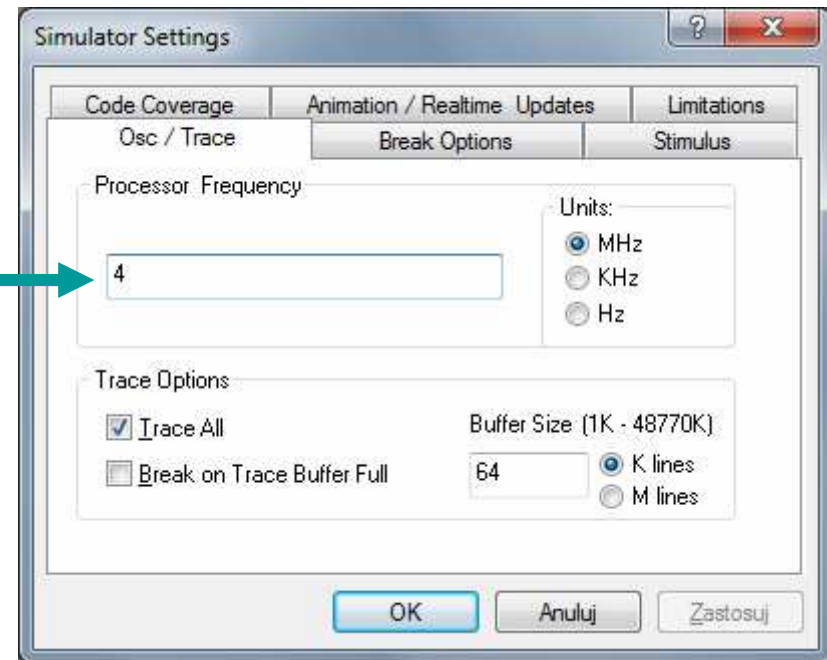
Krok 3. Konfigurowanie symulatora

- Ustawić częstotliwość taktowania mikrokontrolera (dotyczy symulacji w czasie rzeczywistym „realtime”).

Zestaw uruchomieniowy ZL4PIC dostępny w lab. techniki cyfrowej umożliwia tylko:
+ rezonator kwarcowy 4 MHz,
+ wewnętrzny oscylator RC,
np. 8 MHz w układzie PIC16F819.

- Umożliwić podgląd aktualnych wartości rejestrów i pamięci podczas symulacji „realtime”.

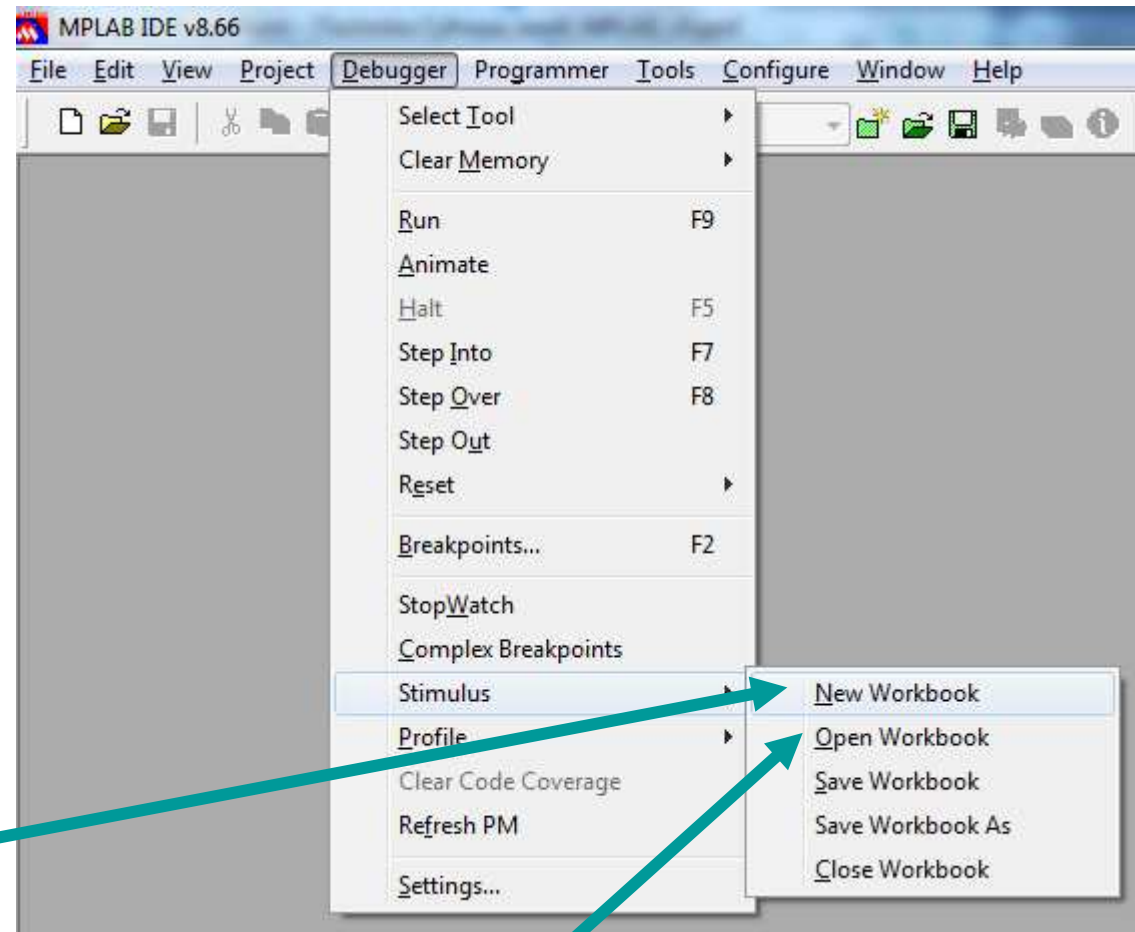
Ta opcja jest domyślnie wyłączona, co uniemożliwia podgląd bieżących wartości rejestrów i pamięci podczas pracy symulatora w czasie rzeczywistym, a zmiany będą pokazane dopiero po wstrzymaniu symulacji (menu „Halt”).



Programowa symulacja mikrokontrolera

Krok 4. Ustawienie wymuszeń na wejściach

➤ Otworzyć nowe puste okno „Stimulus”...



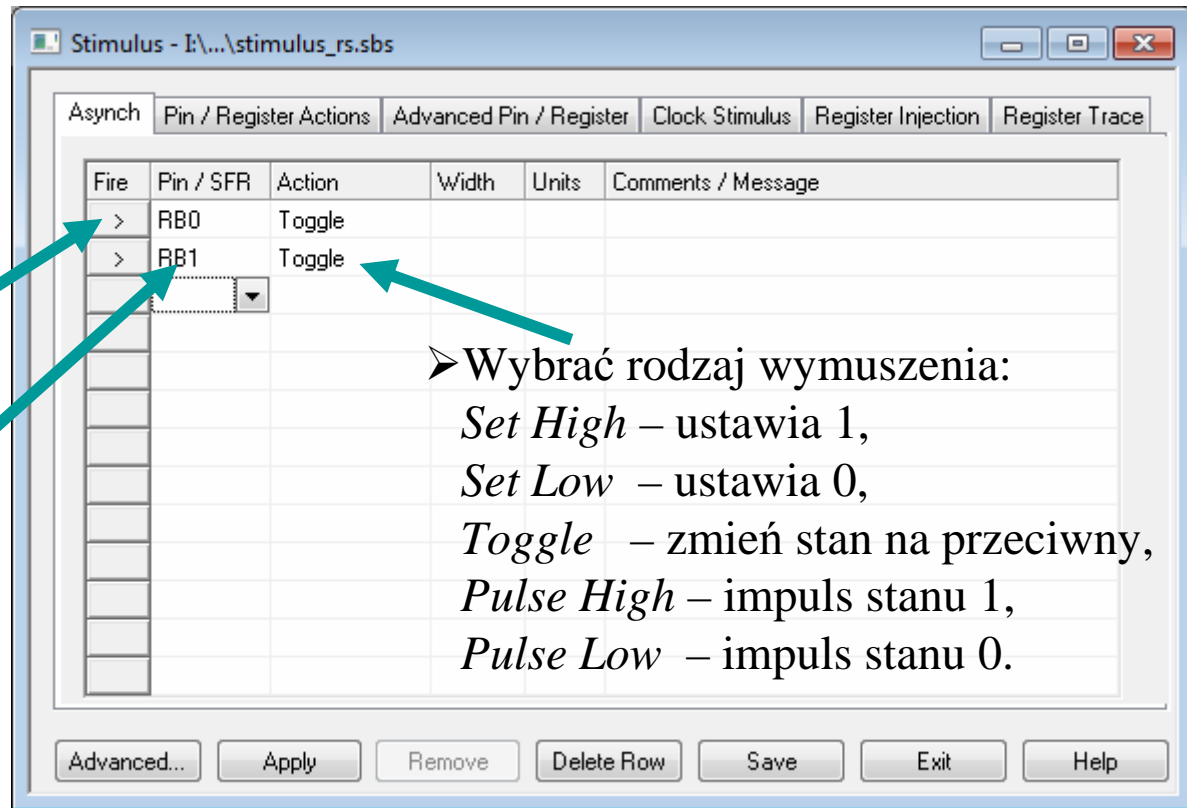
... a jeśli mamy już plik z zapisem stanu okna „Stimulus”, to można go teraz otworzyć.

Programowa symulacja mikrokontrolera

Krok 4. Ustawienie
wymuszeń na wejściach
c.d. ...

Przycisk aktywujący akcję
podczas pracy symulatora

➤ Wybrać odpowiednie
bity w rejestrze PORTB
kontrolującym wej./wyj.



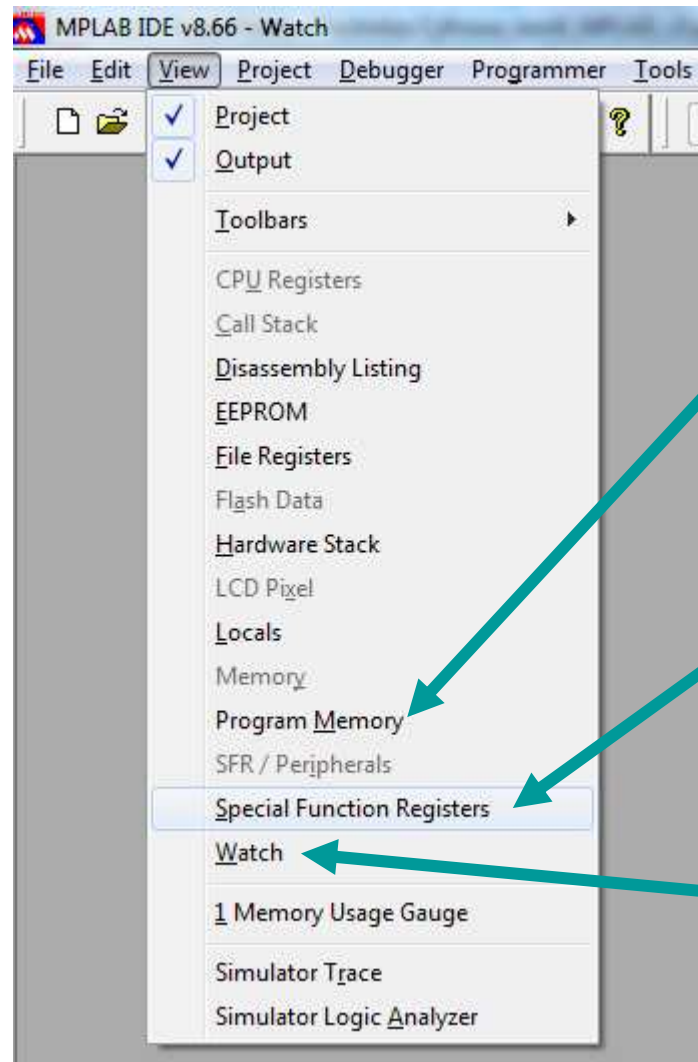
➤ Wybrać rodzaj wymuszenia:
Set High – ustawia 1,
Set Low – ustawia 0,
Toggle – zmień stan na przeciwny,
Pulse High – impuls stanu 1,
Pulse Low – impuls stanu 0.

UWAGA: w celu zachowania zgodności z przyciskami zestawu ZL4PIC (dostępny w pracowni) zaleca się ograniczenie do: RB0 (przycisk S1), RB1 (przycisk S2), RB3 (przycisk S3), RB4 (przycisk S4), RB5 (przycisk S5) – tylko z układem PIC16F819.

UWAGA: przyciski S1...S5 w zestawie ZL4PIC wymuszają 0 podczas ich przytrzymywania. Do ich symulacji najbardziej odpowiednia jest akcja „Toggle”.

Programowa symulacja mikrokontrolera

Krok 5. Otwarcie okien do podglądu wyników działania symulatora

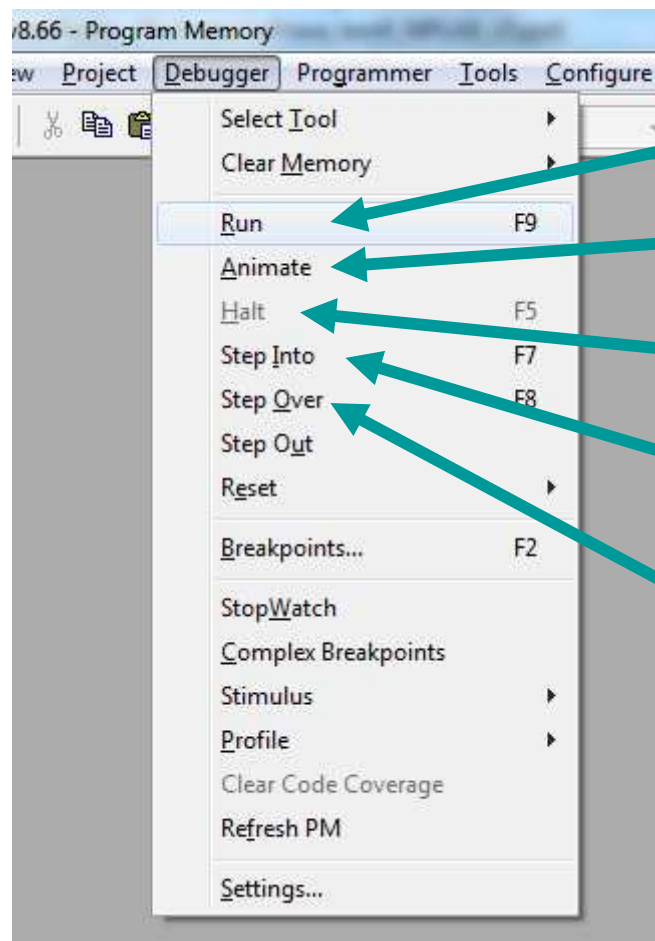


Najczęściej przydające się okna:

- ✓ Podgląd pamięci programu z możliwością translacji na instrukcje asemblera (przydatne przy imporcie z pliku *.hex, bo wtedy nie widzimy źródła programu).
- ✓ Podgląd wszystkich rejestrów mikrokontrolera, w tym 8-bitowe rejestry PORTA i PORTB, których bity RA0, RA1, ..., RB0, RB1, ... kontrolują wejścia/wyjścia dwustanowe.
- ✓ Pokaż tylko rejestry i zmienne wybrane z listy.

Programowa symulacja mikrokontrolera

Krok 6. Uruchomienie symulacji



Sposoby wykonywania programu:

- ✓ Run - symulacja w czasie rzeczywistym.
- ✓ Animate - spowolnione wykonywanie z animacją na oknie instrukcji.
- ✓ Halt - wstrzymanie wykonywania instrukcji,
- ✓ Step Into - wykonywanie krokowe po jednej instrukcji,
- ✓ Step Over - wykonywanie krokowe ale wykonanie procedury wywoływanej instrukcją CALL jest jednym krokiem.

Programowa symulacja mikrokontrolera

Krok 7. Symulacja...

Przyciskami zadajemy stany wybranych wejść

Miejsce animacji lub wstrzymania programu

obserwujemy wyjścia w PORTA

obserwujemy wejścia w PORTB

Tutaj widać tylko wybrane rejestry

Address	SFR Name	Binary
08D	PIE2	00000000
00C	PIR1	00000000
00D	PIR2	00000000
005	PORTA	00000010
006	PORTB	00000011
092	PR2	11111111
093	SSPAD	00000000
013	SSPBUF	00000000
014	SSPCON	00000000
094	SSPSTAT	00000000
003	STATUS	00011100
010	T1CON	00000000
012	T2CON	00000000
001	TMR0	00000000
00E	TMR1	00000000
00F	TMR1H	00000000
00E	TMR1L	00000000
011	TMR2	00000000
085	TRISA	11111100
086	TRISB	11111111
	WREG	00000000

Update	Address	Symbol Name	Value
	005	PORTA	0x02
	006	PORTB	0x03

```
; W pętli wszystkie adresy w banku 0
; Przepisz bity nr 0 i 1 z wyj_Q_nieQ do rej. W z odwróceniem ich
clrw
btfsc  wyj_Q_nieQ,0    ; jeśli ustawiony bit 0,
iorlw  b'10'          ; to ustaw bit 1 w rej. W
btfsc  wyj_Q_nieQ,1    ; jeśli ustawiony bit 1,
iorlw  b'01'          ; to ustaw bit 0 w rej. W

; Symulacja dwóch bramek NAND w przerzutniku RS
```

6.4. Ćwiczenie do samodzielnego wykonania

1). Pobrać przykładowy program z instrukcji do ćw. E58 dostępny na stronie
<https://fizyka.p.lodz.pl/pl/dla-studentow/fizyka-tech/tc/>

Program realizuje funkcję przerzutnika RS z bramek NAND zapisywanego poziomem 0.

2). Otworzyć program w pakiecie MPLAB IDE.

3). W oknie „Select Device” wybrać urządzenie PIC16F819.

4). W oknie „Configuration Bits” zaznaczyć „Configuration Bits set in code”.

5). Skompilować program.

6). Uruchomić symulację w MPLAB SIM i sprawdzić, czy program prawidłowo realizuje funkcję przerzutnika RS zapisywanego poziomem 0 o następujących wyprowadzeniach:

RB0 – wejście przerzutnika $\sim R$ (ang. Reset) aktywne po podaniu 0,

RB1 – wejście przerzutnika $\sim S$ (ang. Set) aktywne po podaniu 0,

RA0 – wyjście przerzutnika Q,

RA1 – wyjście przerzutnika $\sim Q$.

RA0 i RA1 są bitami numer 0 i 1 w 8-bitowym rejestrze PORTA,

RB0 i RB1 są bitami numer 0 i 1 w 8-bitowym rejestrze PORTB.

6.5. Literatura

- [1] T. Jabłoński, *Mikrokontrolery PIC16F8x w praktyce*, Wydawnictwo BTC, Warszawa 2002.
- [2] S. Pietraszek, *Mikroprocesory jednoukładowe PIC*, Helion, Gliwice 2002.
- [3] T. Jabłoński, K. Pławsiuk, *Programowanie mikrokontrolerów PIC w języku C*, Wydawnictwo BTC, Warszawa 2005.
- [4] Instrukcja do ćwiczenia laboratoryjnego E58 „Wprowadzenie do programowania mikrokontrolerów PIC16 w języku assembler”, dostępna na stronie <https://fizyka.p.lodz.pl/pl/dla-studentow/tc/>
- [5] Dokumentacja wewnątrz pakietu MPLAB IDE, menu Help.