**İ.T.Ü.**

**Faculty of Computer and Informatics**

**Computer Engineering**



**MICROCOMPUTER LAB**

**REPORT**

|  |  |  |
| --- | --- | --- |
| **Lab No** | **:** 05 | |
| **Lab Date** | **:** 31.10.2013 | |
| **Group** | **:** 09 | |
| **Group Members** | **:** 040100014 Teoman Turan  040100018 Mustafa Durmuş  040100117 Tuğrul Yatağan  040100124 Emre Gökrem | |
|  |  | |
|  |  | |
|  |  | |
| **Research Assistant :** Figen Şentürk | | |  |
|  | |  |

**1. THE AIM/CONTENT of THE EXPERIMENT**

This is an experiment to introduce us CSM12C32 Kit. It is done by creating a project that is constructed from a code in Assembly and C programming languages via CodeWarrior software.

**2. EQUIPMENT**

To write and compile the code on CodeWarrior software, one of the laboratory’s desktop computers has been used. CSM12CS32 Kit has been connected to that computer. After the compilation of the code, its result has been observed on the kit.

**3. EXPERIMENTS**

**3.1. Section 2: Assembly and C Project Creation**

CodeWarrior software had already been installed on the laboratory’s desktop computers. For the basic setup, we have followed the steps demonstrated with screenshots on the experiment sheet. The example code coming along in *main.asm* calculates Fibonacci numbers. We have cleared that Assembly source code file and pasted new Assembly code which can be found on the page after where the last step is explained on the experiment sheet, then clicked on Debug button. For the connection and compilation setup, we have followed the rest of the steps there.

As can be seen from the comments there, PORTA.0 of the kit has been connected to LED1, and it has become an output. Then, PORTB.4 of the kit has been connected to LED2, and it has become another output. Afterward, LED1 has been off by making PORTA.0 loaded with $01, and LED2 has been off by making PORTB.4 loaded with $10. So, initially, both LEDs are off. Also, at the end of the code, there is a code fragment that causes delays between two LEDs.

Within a loop including a small delay, two LEDs on the kit blink respectively as the result.

**3.2. Section 3: Experiment**

For this experiment, the programming language option of CodeWarrior has been changed from Assembly to C in the relevant step of the basic setup for a new project. Also, connection type has been changed to HCS12 Serial Monitor in the relevant step of the setup. The rest are the same. Afterward, the following C code has been run:

#include <hidef.h> /\* common defines and macros \*/

#include <mc9s12c32.h> /\* derivative information \*/

#pragma LINK\_INFO DERIVATIVE "mc9s12c32"

byte varX;

void delayS() **/\* function to provide delay between two LEDs \*/**

{

word i,j;

for(i=0; i<0x0010; i++)

for(j=0; j<0xFFFF; j++){}

}

void main(void)

{

**/\* The connections between PORTA.0-LED1 and PORTB.4-LED2 are established here with the following assignments as done before in the Assembly code. \*/**

DDRA=0x01; //PORTA.0 connected to LED1 is output

DDRB=0x10; //PORTB.4 connected to LED2 is output

PORTA=0x00; //LED1 is on **/\* Initially both LEDs are on. \*/**

PORTB=0x00; //LED2 is on

varX=0x11;

for(;;) **/\* LEDs blink respectively in an infinite loop. \*/**

{

PORTA=varX; **/\* assignments for respective blinking \*/**

PORTB=varX;

varX=~varX;

delayS(); **/\* Function to provide delay is called. \*/**

}

}

In addition, the Assembly code of the experiment is the following one:

; export symbols

XDEF Entry ; export 'Entry' symbol

ABSENTRY Entry ; for absolute assembly: mark this as application entry point

; include derivative specific macros

INCLUDE 'mc9s12c32.inc'

ROMStart EQU $4000 ; absolute address to place my code/constant data

; variable/data section

ORG RAMStart

Counter1 DC.W $FFFF

Counter2 DC.W $0010

; code section

ORG ROMStart

Entry:

LDS #RAMEnd+1 ; initialize the stack pointer

LDAA #$01

STAA DDRA ; PORTA.0 connected to LED1 is output

LDAA #$10

STAA DDRB ; PORTB.4 connected to LED2 is output

LDAA #$01

STAA PORTA ; PORTA.0 is 1, LED1 is off

LDAA #$10

STAA PORTB ; PORTB.4 is 1, LED2 is off

loop: COMA

STAA PORTA

STAA PORTB

JSR delayS

BRA loop

delayS: LDY Counter2

delaySloop: JSR delayMs

DEY

BNE delaySloop

RTS

delayMs: LDX Counter1

delayMsloop: DEX

BNE delayMsloop

RTS

As the result, within an infinite loop including a delay, two LEDs blink respectively.