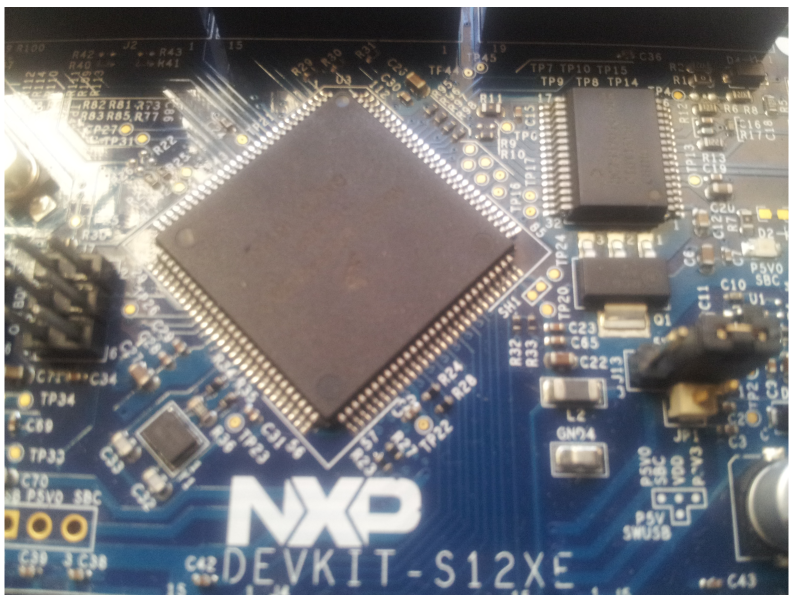
The MC-II Electronic Calculator

An Embedded Systems Project



By Danny Milutinovic
Melbourne, Australia
November 3, 2018

Contents

1 Introduction

The MC-II is a programmable, reverse polish notation scientific and statistical calculator. This paper describes how to build the MC-II using a low cost development board, graphical liquid crystal display and matrix keypad, as well as the algorithms used to implement the mathematical functions. Each chapter builds on the material covered in the previous one so that the code is built in stages, making it easier to follow. The source code for each stage is written in assembly language and is available at `https://github.com/DanielMilutinovic/MC-II`.

2 Getting Started

2.1 Hardware

The following is a list of the hardware components required to build the MC-II. They can be purchased from electronic components distributors such as Mouser, Digi-Key and element14:

1. The DEVKIT-S12XE development board by NXP (Figure 1). It features the 9S12XEP100 microcontroller and is provided with a USB cable to connect it to a computer.

Figure 1: DEVKIT-S12XE

2. An ST7565 128x64 graphical liquid crystal display (GLCD) (Figure 2). The model shown is available from Core Electronics and Adafruit. Note that eleven 0.1" (2.54mm) male header pins must be soldered to the board if using this particular model; however it is simple through-hole soldering.

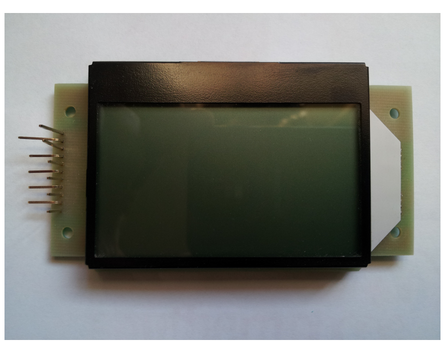


Figure 2: ST7565 GLCD

3. 0.1” (2.54mm) male-female jumper cables (Figure 3) to interface the DEVKIT-S12XE development board to the ST7565 GLCD and matrix keypad.

Figure 3: 0.1” male-female jumper cables

4. A matrix keypad. Standard 4x4 matrix keypads (Figure 4) work well but an 8x6 keypad (Figure 5) allows more functions to be added easily. Interfacing to each type of keypad is covered in Chapter 4. The 8x6 matrix keypad shown was designed by Dirk Heisswolf and manufactured by Seeed Studio. I have several 8x6 matrix keypad PCB’s left (contact me at `daniel_milutinovic@yahoo.com.au`) and the files are available at `https://github.com/DanielMilutinovic/MC-II` if you would like to order them (Seeed Studio has a minimum order of 10 PCB’s). If making the 8x6 matrix keypad you will need to solder 0.1” (2.54mm) male header pins and 44 push buttons to the PCB. I used Omron 6x6 tactile switches, part no. B3F-1052.

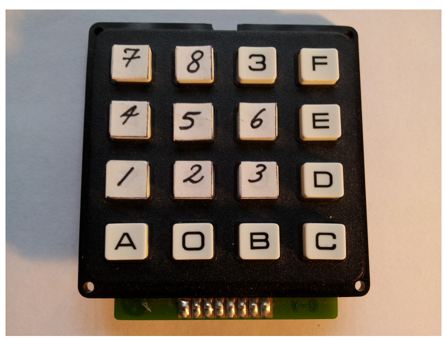


Figure 4: 4x4 matrix keypad by EOZ, with some key labels attached

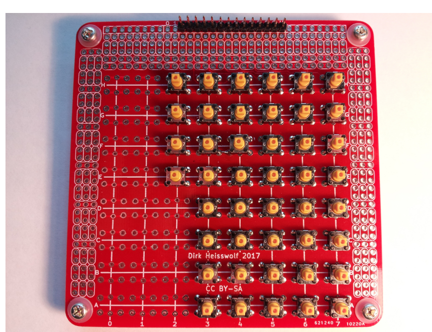


Figure 5: Custom 8x6 matrix keypad

2.2 Software

The 9S12XEP100 microcontroller on the DEVKIT-S12XE development board is programmed using the CodeWarrior development software by NXP. CodeWarrior may be downloaded free of charge and there are no code size limitations for assembly language files. It features a debugger and a convenient simulation mode, which enables code to be tested without downloading it to the actual microcontroller, thereby saving the Flash memory on the device from repeated writes.

2.3 Blinking an LED

Install "CodeWarrior Development Studio for the S12(X) Version 5.2" onto your computer. Connect the DEVKIT-S12XE development board to your computer using the supplied USB cable, then connect a power supply to the development board (a 12V power supply is recommended but a 9V power supply works). The LED on the development board will turn on and change colour (the board comes with a pre-loaded example project - see the DEVKIT-S12XE Quick Start Guide for further information, which can be downloaded at `https://github.com/DanielMilutinovic/MC-II`).

We will now download a new program to the board to blink the LED. Firstly create a new folder on your computer to save your projects to, then open CodeWarrior and choose "Create New Project". Select "MC9S12XEP100" and "P & E USB BDM Multilink". Then press "Next" (Figure 6).

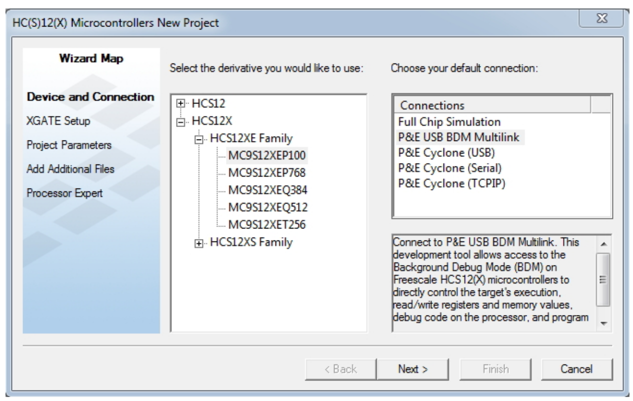


Figure 6: Selecting the microcontroller and connection

Select "Single Core(HCS 12X)" then press "Next" (Figure 7).

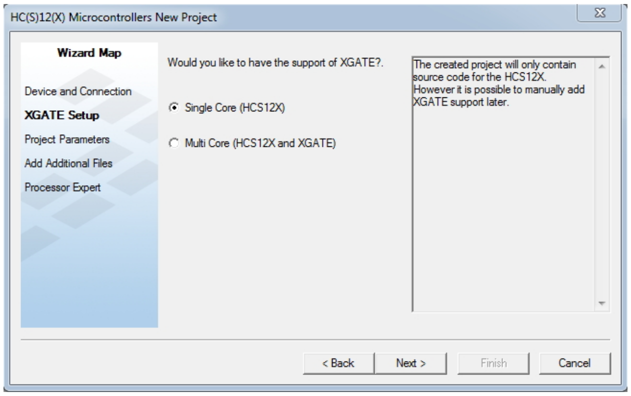


Figure 7: Selecting the core

Select "Absolute assembly" and enter the location and project name ("LED" in this example), then select "Finish" (Figure 8).

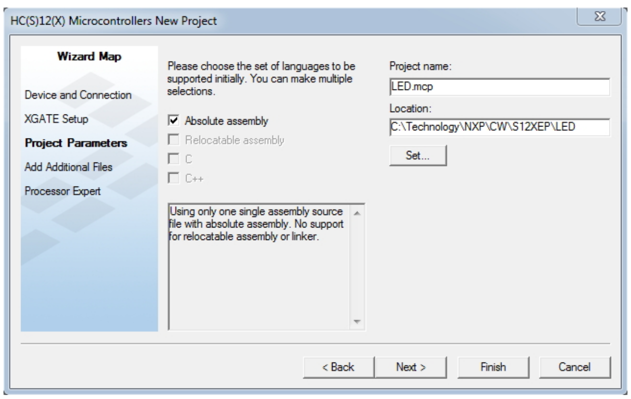


Figure 8: Choosing the language, name and location

Delete all of the pre-filled assembly code that now appears in the right-hand window. Go to `https://github.com/DanielMilutinovic/MC-II` and open the file LED.asm (an assembly language program to flash the LED). Copy all of the text that appears and paste it to where the pre-filled assembly code used to be. Then press "Make" in the tool bar to assemble the file (Figure 9).

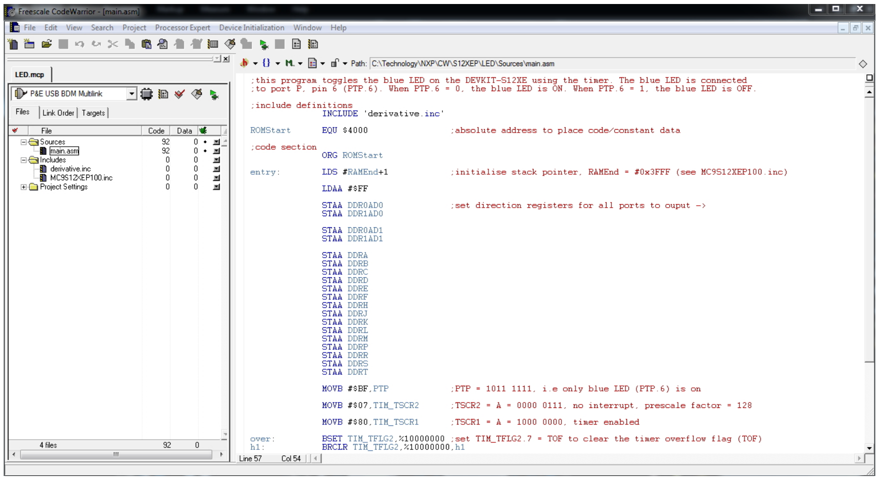


Figure 9: Assembling the file

Now press "Debug" in the tool bar to download the code to the microcontroller and then press "Start / Continue" in the new window to run the program. The blue LED on the board will start to blink. You are now ready to interface the DEVKIT-S12XE to the ST7565 GLCD.

3 Interfacing the NXP DEVKIT - S12XE development board to the ST7565 GLCD

WARNING! Ensure the jumper at J13 on the DEVKIT-S12XE development board is set to 3.3V. Otherwise the ST7565 GLCD will be permanently damaged.

Figure 10 shows the wiring diagram for interfacing the DEVKIT-S12XE development board to the ST7565 GLCD (without a backlight). Refer to the DEVKIT-S12XE Quick Start Guide for the location of the pins on the development board and use the male-female jumper cables to connect the DEVKIT-S12XE to the ST7565 GLCD.

3.1 Displaying the status line using simulated SPI

Create a new project, copy and paste the code from GLCD1.asm, assemble the file, download and run. The display should appear as in Figure 11.

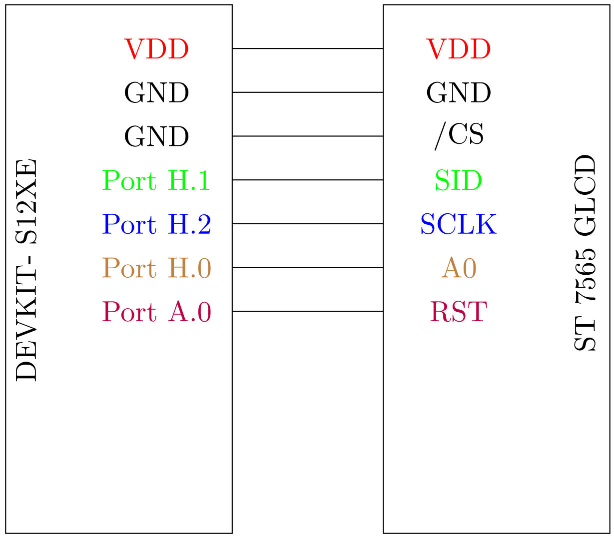


Figure 10: Interfacing the DEVKIT-S12XE to the ST 7565 GLCD

Figure 11: The status line

3.2 Displaying the status line, stack level labels and menu using SPI

Create a new project, copy and paste the code from GLCD2.asm, assemble the file, download and run. The display should appear as in Figure 12.

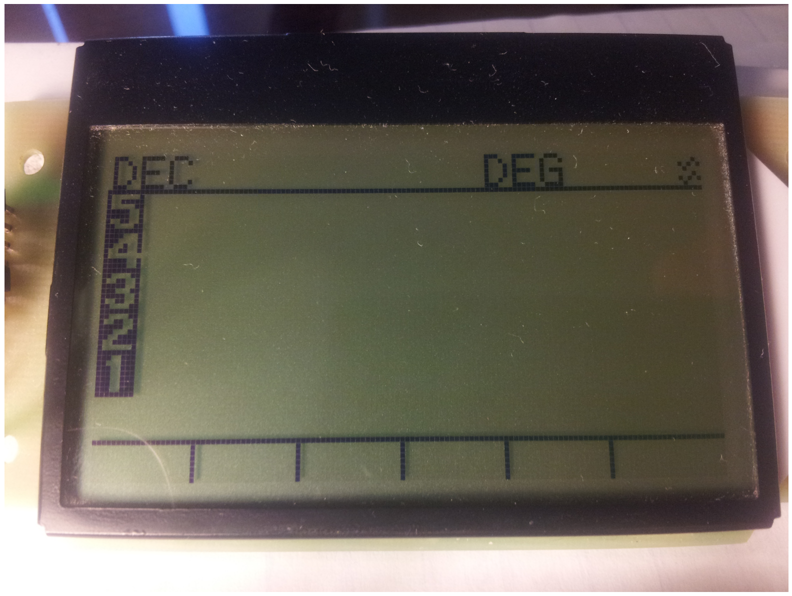


Figure 12: The status line, stack level labels and menu (currently empty)

3.3 Displaying the status line, stack level labels, command line cursor and menu using SPI and interrupts

GLCD3.asm completes the display by adding the blinking cursor to the command line. The display is now divided into four sections, as shown in Figure ?

You are now ready to interface the DEVKIT-S12XE to a matrix keypad.

4 Interfacing the NXP DEVKIT - S12XE to a matrix keypad

4.1 4x4 matrix keypad

Figure 13 shows the functions on the 4x4 matrix keypad.

Figure 14 shows the wiring diagram for interfacing the DEVKIT-S12XE development board to the 4x4 matrix keypad. Refer to the DEVKIT-S12XE Quick Start Guide for the location of the pins on the development board and use the male-female jumper cables to connect the DEVKIT-S12XE to the 4x4 matrix keypad.

4.2 8x6 matrix keypad

Figure 15 shows the functions on the 8x6 matrix keypad.

Figure 16 shows the wiring diagram for interfacing the DEVKIT-S12XE development board to the 8x6 matrix keypad. Refer to the DEVKIT-S12XE Quick Start Guide for the location of the pins on the development board and use the male-female jumper cables to connect the DEVKIT-S12XE to the 8x6

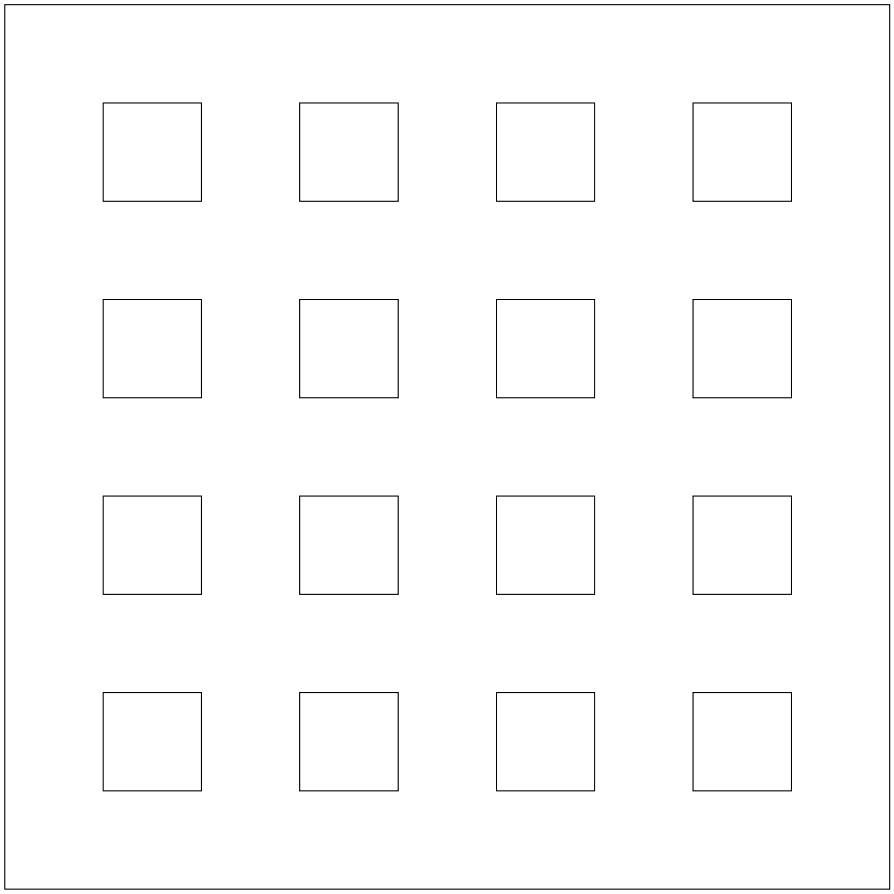


Figure 13: The 4x4 matrix keypad functions

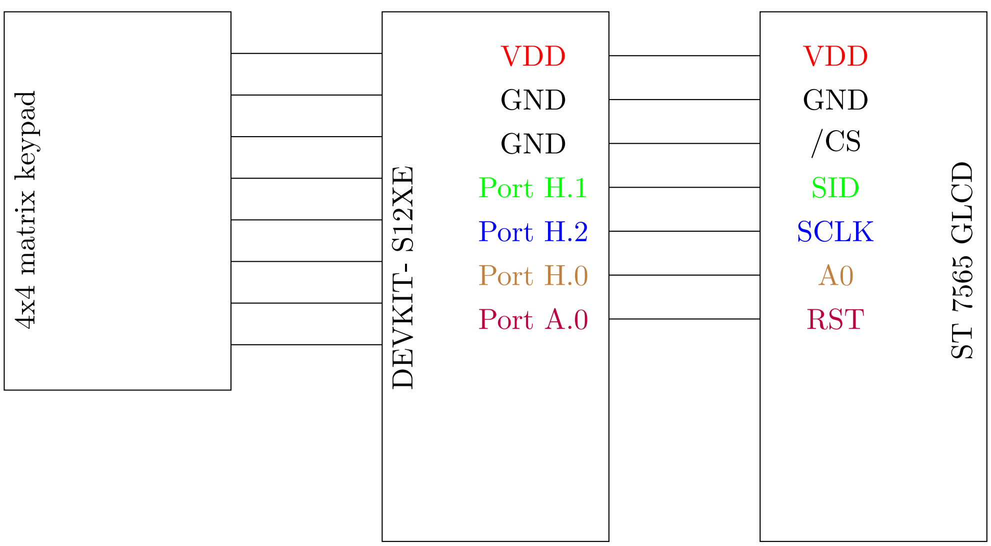


Figure 14: Interfacing the DEVKIT-S12XE to the 4x4 matrix keypad

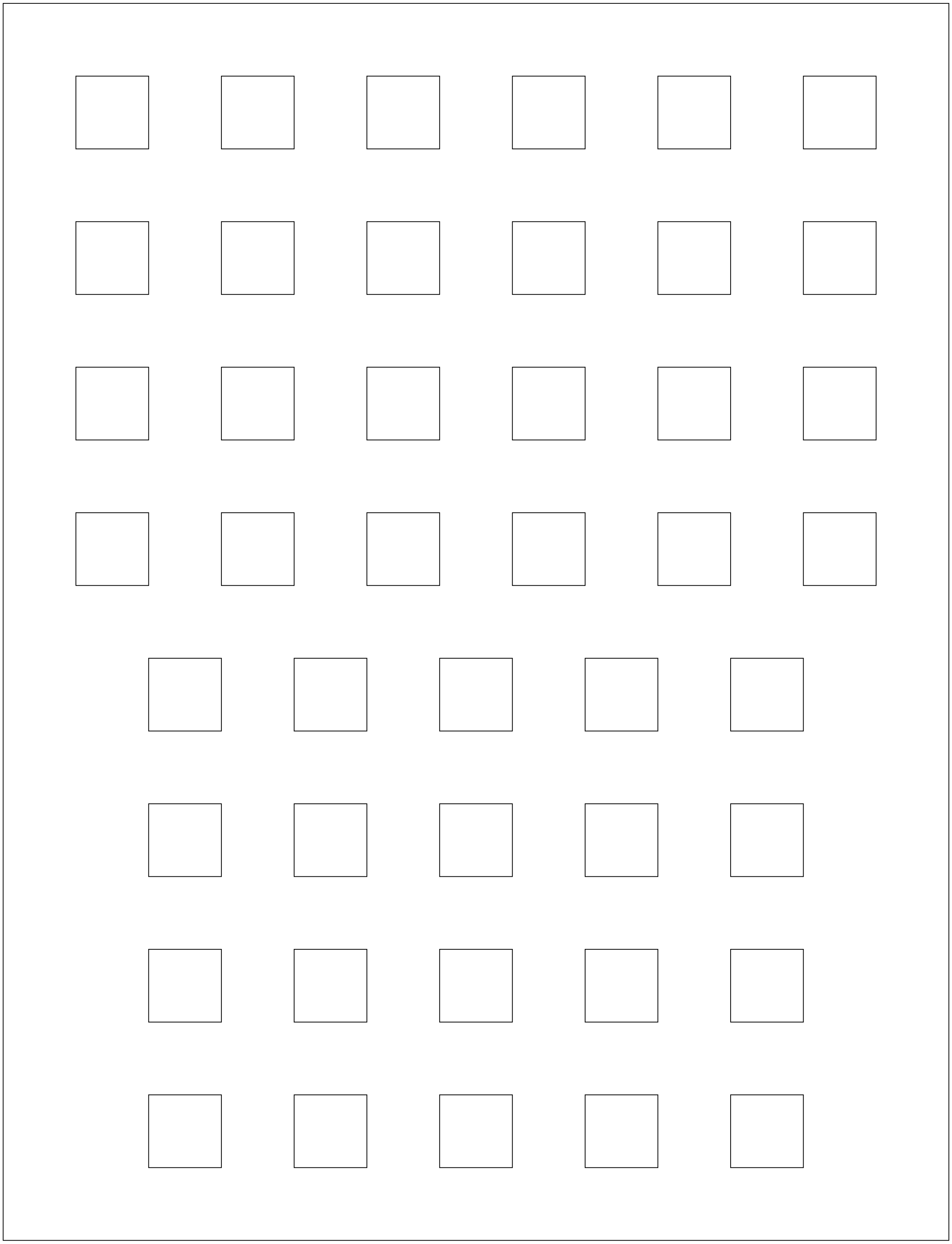


Figure 15: The 8x6 matrix keypad functions

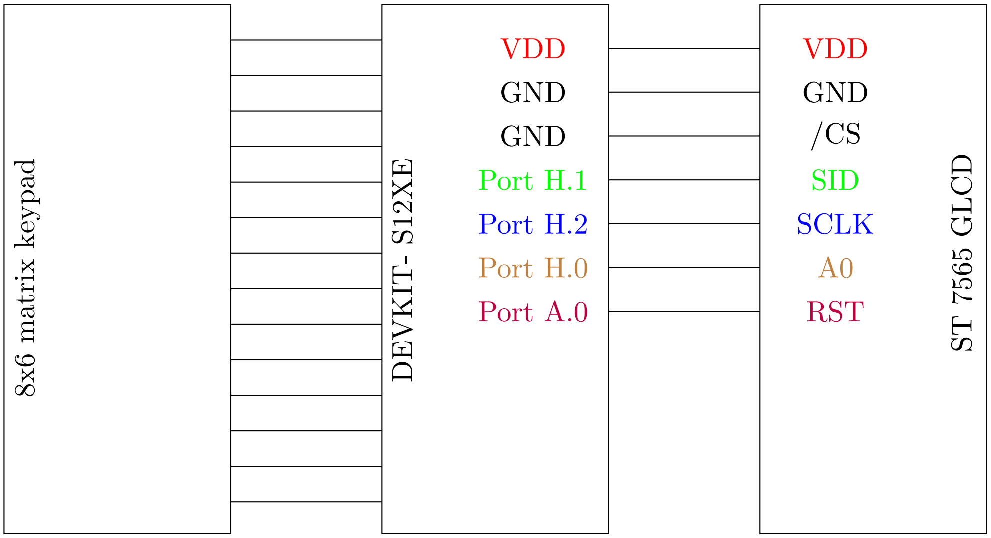


Figure 16: Interfacing the DEVKIT-S12XE to the 8x6 matrix keypad

matrix keypad.

5 The command line and stack

5.1 Entering values on the command line

Note: from this point onward it is assumed that the DEVKIT-S12XE is interfaced to the 8x6 matrix keypad. Modifications to the code will need to be made if it is interfaced to the 4x4 matrix keypad.

5.2 Pushing values onto the stack

6 The stack operations

6.1 Enter

6.2 Delete and Clear

6.3 Pick

6.4 Rotate and Unrotate

6.5 Over

7 The functions negation, addition, subtraction, multiplication and division

7.1 Negation ($\pm$)

7.2 Addition ($+$)

7.3 Subtraction ($-$)

7.4 Multiplication ($\times$)

There are three common algorithms used to implement multiplication: shift and add, long multiplication using the assembly instruction emul and a CORDIC method.

7.5 Division ($\div$)

As with multiplication, there are three common algorithms used to implement division: shift and subtract, long division using the assembly instruction ediv and a CORDIC method.

8 Memory

8.1 Storing and recalling values

8.2 Viewing the memory contents of stack level 1

It is convenient to be able to view the memory contents of stack level 1 when greater accuracy is required as well as for debugging purposes.

8.3 Viewing execution time

In order to compare the efficiency of different algorithms the microcontroller's timer may be used to obtain an approximation to the number of machine cycles required to execute a particular function.

9 The scientific functions

9.1 Square (x^2)

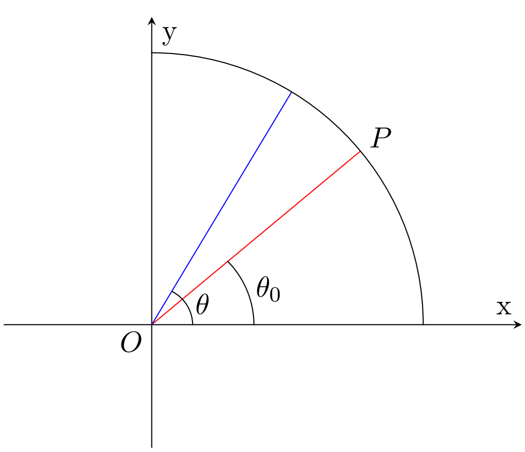
9.2 Factorial ($x!$)

9.3 Square root ($\sqrt{x}$)

9.4 Sine, Cosine and Tangent ($\sin\theta$, $\cos\theta$ and $\tan\theta$)

Firstly we look at the CORDIC algorithm for $\tan\theta$. Suppose we want to find $\tan\theta$, where θ is in radians, and suppose

$$\tan\theta_0 = \frac{Y}{X}, \text{ where } X, Y \text{ are known integers and } \theta_0 < \theta$$



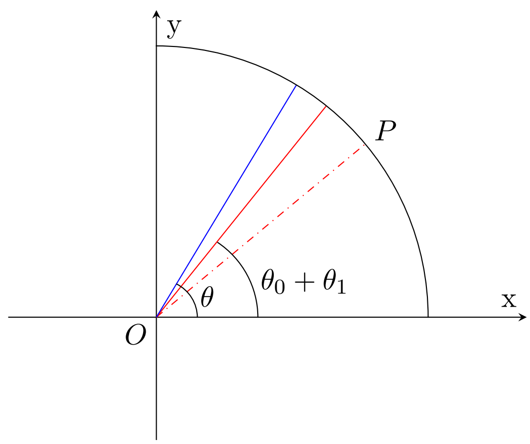
$\frac{Y}{X}$ is then an approximation to $\tan\theta$. To obtain a better approximation we rotate line OP anticlockwise

θ_1 radians, where θ_1 is chosen so that

$$\tan\theta_1 = 10^{-K}, \text{ for some } K = 0, 1, 2, ..., 13$$

and

$$\theta_0 + \theta_1 \leq \theta$$



Note that $\tan(\theta_0 + \theta_1)$ is a better approximation to $\tan\theta$.

We know use the trigonometric identity

$$\tan(\alpha + \beta) = \frac{\tan\alpha + \tan\beta}{1 - \tan\alpha\tan\beta}$$

to obtain

$$\tan(\theta_0 + \theta_1) = \frac{\tan\theta_0 + \tan\theta_1}{1 - \tan\theta_0\tan\theta_1}$$

To find $\sin\theta$, $\tan\theta$ is found using the CORDIC algorithm and the result substituted into the trigonometric identity

$\sin\theta$

9.5 Inverse Sine, Inverse Cosine and Inverse Tangent ($\sin^{-1} x$, $\cos^{-1} x$ and $\tan^{-1} x$)

9.6 The exponential functions e^x and 10^x

9.7 The logarithmic functions $\ln x$ and $\log_e x$

9.8 The exponential functions x^y and $\sqrt[y]{x}$

9.9 The random number generator

10 Keystroke programming

11 Binary and hexadecimal numbers

12 Complex numbers

13 Rational numbers

14 The statistical functions

15 Plotting functions

16 Matrices

17 The triangle solver

18 Building a hand-held MC-II