SIEMENS

Ingenuity for life

FPGA **Digital Track** Training

Final Project: Implementing 3x3 **Matrix multiplier** using different implementations

(**FSMD**, **NIOS II SoC**, and **ARM Cortex M0**)

Submitted To:

Dr. Ihab Adly

Submitted By:

1. Ahmed Adel Abd Elmonem
2. Shady Shokri Ramzy
3. Mostafa Hassanien Ahmed

Table of Contents

1 FSMD 3
2 NIOS II SoC 5
3 ARM Cortex M0 6
4 Observation 6
5 Appendices 7
5.1 NIOS II C Code 7
5.2 ARM Cortex M0 C Code 8

Table of Figures

Figure 1: FSM RTL Schematic 3
Figure 2: Datapath RTL Schematic 4
Figure 3: Functional Simulation 4
Figure 4: Timing Simulation 4
Figure 5: NIOS II Microcontroller Schematic 5
Figure 6: Timing Simulation 5
Figure 7: ARM Cortex M0 Functional Simulation 6

Tables

Table 1: Comparison between different implementations: FSMD, NIOS II, and Cortex M0 6

1 FSMD

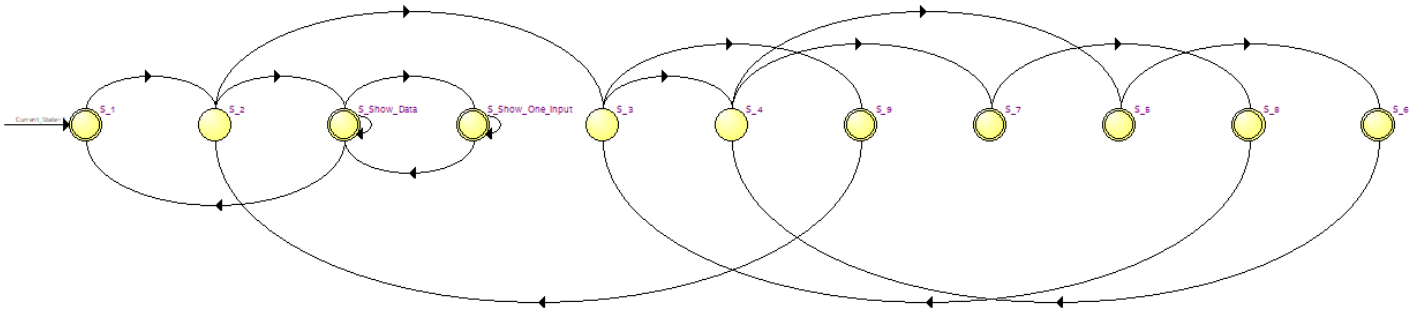

Figure 1: State Machine Diagram

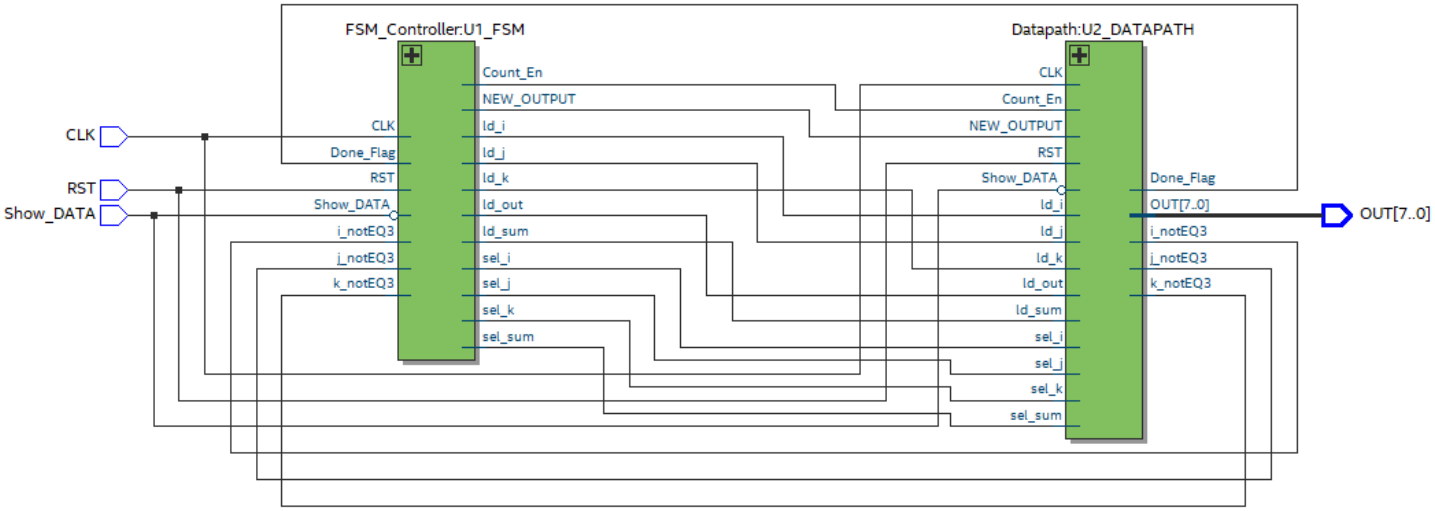

Figure 2: Top Level RTL Schematic

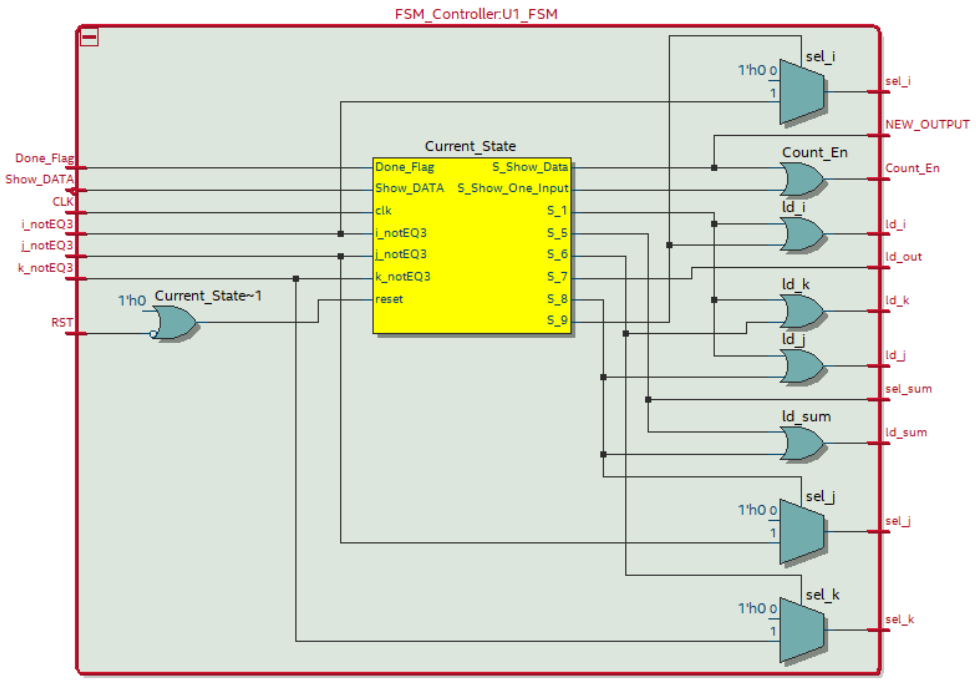

Figure 1: FSM RTL Schematic

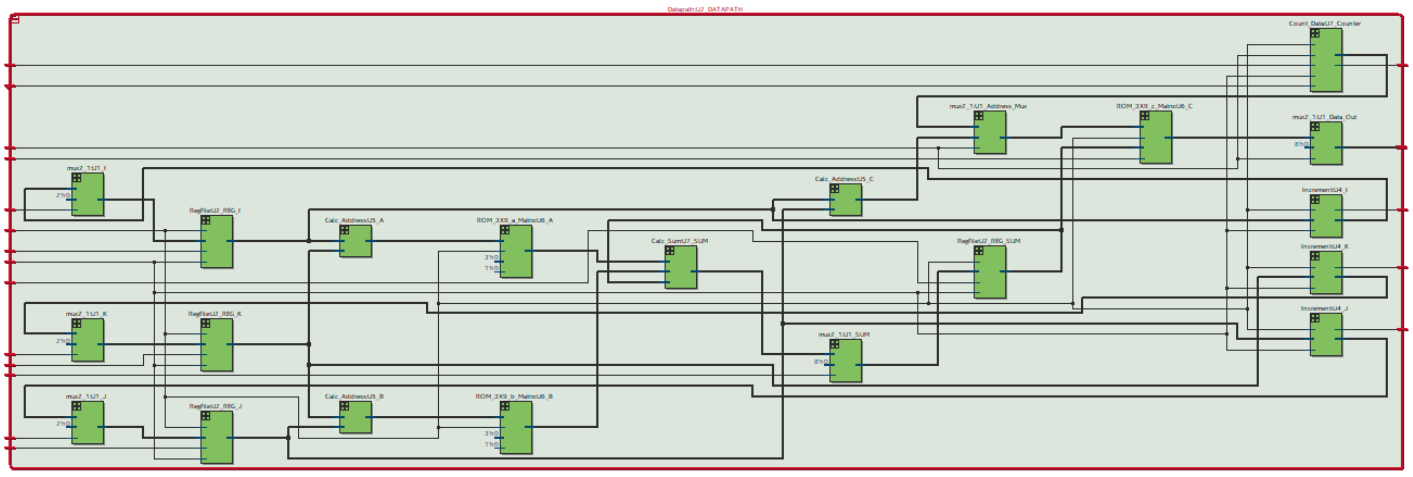

Figure 2: Datapath RTL Schematic

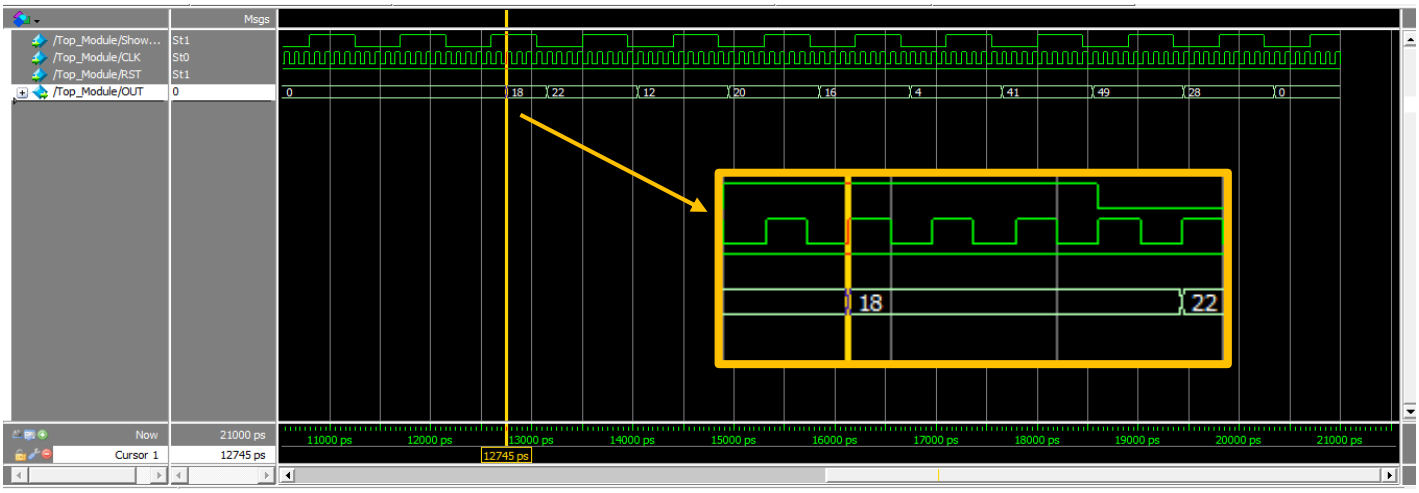

Figure 3: Functional Simulation

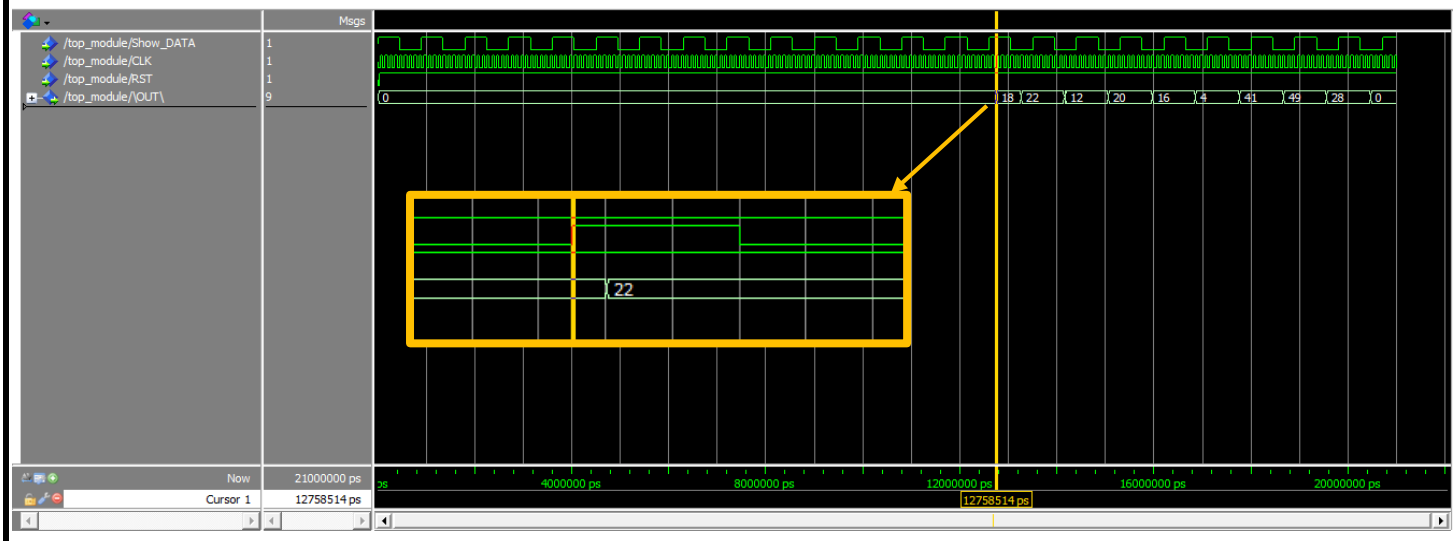

Figure 4: Timing Simulation

2 NIOS II SoC

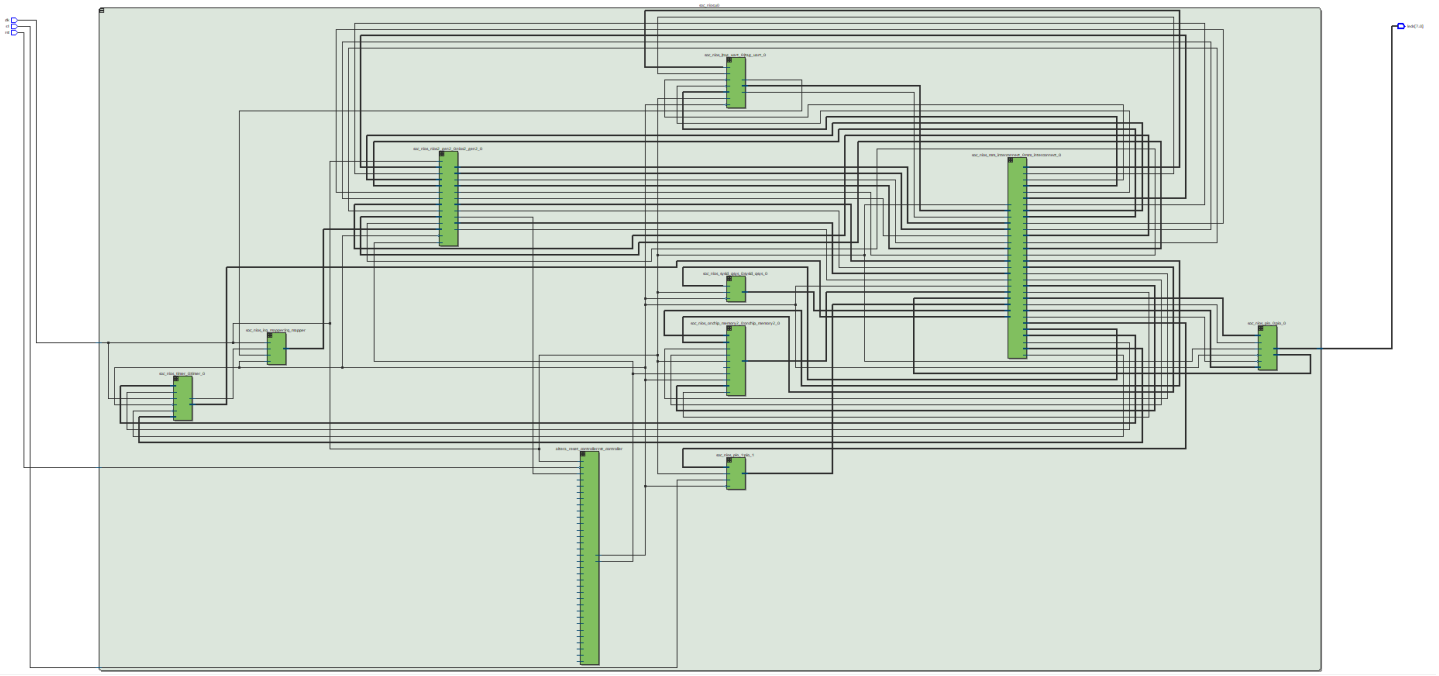

Figure 5: NIOS II Microcontroller Schematic

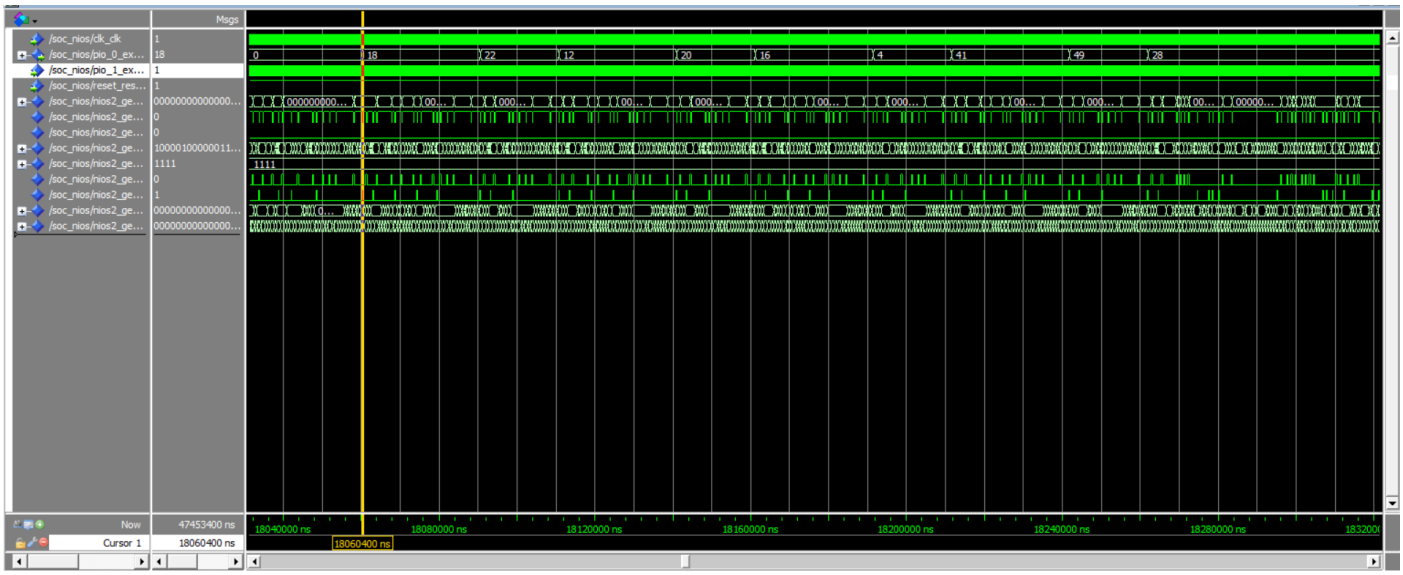

Figure 6: Timing Simulation

3 ARM Cortex M0

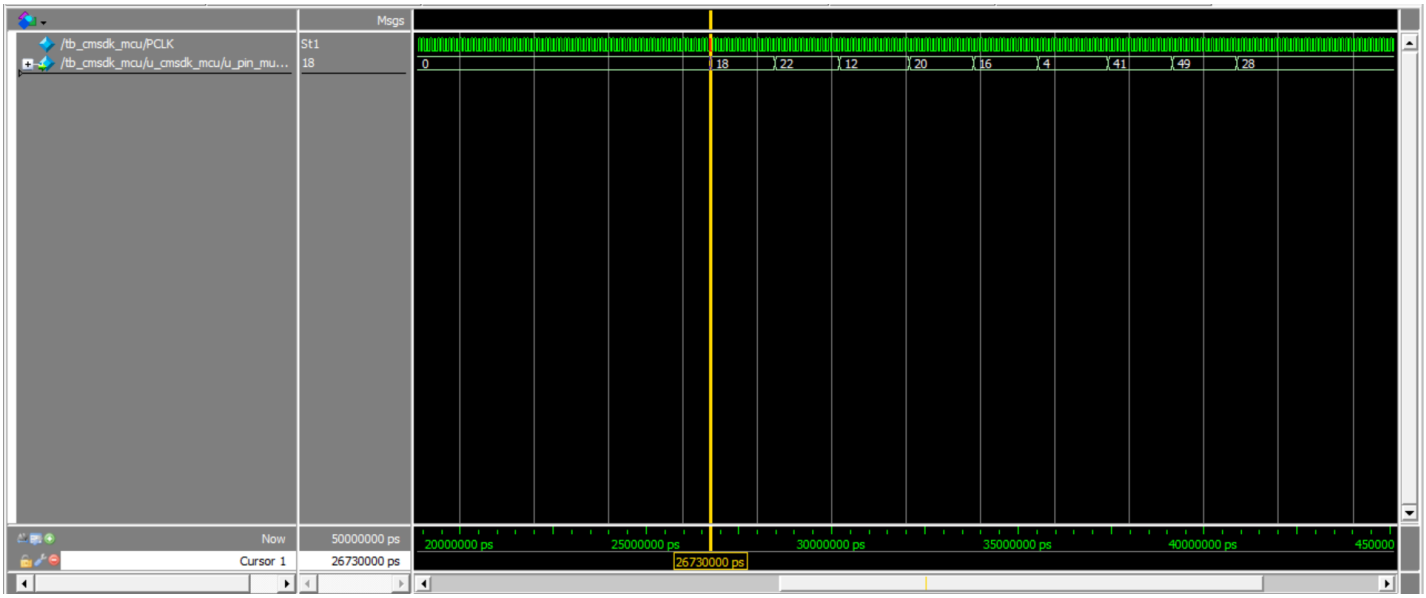

Figure 7: ARM Cortex M0 Functional Simulation

4 Observation

	FSMD	NIOS II	Cortex M0
Number of Cycles	128	180, 604	268

Table 1: Comparison between different implementations: FSMD, NIOS II, and Cortex M0

Obviously, FSMD is the fastest implementation regarding execution time followed by ARM Cortex M0. In contrast, NIOS II is the slowest implementation.

5 Appendices

5.1 NIOS II C Code

```
# include <system.h>
# include <altera_avalon_pio_regs.h>
#include <stdio.h>

int main (void){
    //inputing the two 3x3 matrices
    int a[9] = {1,2,3,5,0,1,2,3,7};
    int b[9] = {3,2,0,0,1,0,5,6,4};

    int c[9];

    int en = 1;

        int i;
        int j;
        int x;



        //multiplication loops
    for( i = 0;i < 3;i++){
        for( j = 0;j < 3;j++){
            c[3*i + j] = 0;
            for( x = 0;x < 3;x++){
                 c[3*i + j] = c[3*i + j] + (a[3*i + x] * b[j + 3*x]);
            }
        }
    }

    for ( int k = 0; k < 9; ++ k )
    {
      en = IORD_ALTERA_AVALON_PIO_DATA(PIO_1_BASE);
      while(en)
      {
          en = IORD_ALTERA_AVALON_PIO_DATA(PIO_1_BASE);
      }
      if ( en == 0 )
      {
        IOWR_ALTERA_AVALON_PIO_DATA(PIO_0_BASE, c[k] );
      }
    }
    return 0;
}
```

5.2 ARM Cortex M0 C Code

```
#ifdef CORTEX_M0

#include "CMSDK_CM0.h"
#include "core_cm0.h"
#endif

#ifdef CORTEX_M0PLUS
#include "CMSDK_CM0plus.h"
#include "core_cm0plus.h"
#endif

#include <stdio.h>

int main (void){
    //inputing the two 3x3 matrices
    int a[9] = {1,2,3,5,0,1,2,3,7};
    int b[9] = {3,2,0,0,1,0,5,6,4};

    int c[9];

        int i;
        int j;
        int x;

        //initializing GPIO0
        CMSDK_GPIO0->OUTENABLESET = 0xFFFF;
        CMSDK_GPIO0->ALTFUNCCLR = 0xFFFF;

        //multiplication loops
    for( i = 0;i < 3;i++){
        for( j = 0;j < 3;j++){
            c[3*i + j] = 0;
            for( x = 0;x < 3;x++){
                 c[3*i + j] = c[3*i + j] + (a[3*i + x] * b[j + 3*x]);
            }
 //outputing value of the array element on LEDs connected to GPIO0 first 9 pins
                        CMSDK_GPIO0->DATA = c[3*i + j];
        }
    }

    return 0;
}
```