Εργαστήριο 10

Ονομα: Μαυρογιώργης Δημήτρης
ΑΜ: 2016030016
ΗΡΥ 411 - Ενσωματωμένα Συστήματα Μικροεπεξεργατών
Πολυτεχνειο Κρητης

April 19, 2021

Σκοπός

Σκοπός του δέκατου εργαστηρίου είναι να κατανοήσουμε τις υπολογιστικές δυνατότητες ενός μικροελεγκτή AVR κάνοντας ένα απλό πρόγραμμα πολλαπλασιασμού πινάκων 3x3 με δύο διαφορετικού τρόπους: 1) χρησιμοποιώντας ως τύπο δεδομένων integers και 2) χρησιμοποιώντας ως τύπο δεδομένων floats.

Περιγραφή της υλοποίησης

Αρχικά, δηλώθηκαν 6 arrays, εκ των οποίων τα 3 είναι τύπου int και τα άλλα 3 είναι τύποθ float. Πιο συγκεκριμένα για κάθε τύπο δεδομένων έχουμε τους 2 πίνακες που πολλαπλασιάζουμε και έναν τρίτο στο οποίο αποθηκεύουμε το αποτέλεσμα. Ολα τα δεδομένα των πινάκων, καθώς και τα αποτελέσματα του πολλαπλασιασμού βρίσκονται στημνήμη του AVR.

Η αρχικοποίηση των πινάκων που πολλαπλασιάζουμε γίνεται κατά τη δήλωση των global arrays. Επιπλέον για τον πολλαπλασιασμό των πινάκων δημιουργήθηκαν δύο ίδιες ρουτίνες με μοναδική διαφορά ότι η πρώτη κάνει το πολλαπλασιασμό των int arrays, ενώ η δεύτερη κάνει τον πολλαπλασισμό των float arrays. Η υλοποίησή τους είναι απλή και με ένα τριπλό βρόγχο επανάληψης υπολογίζουμε μέσω πολλαπλασιασμών και προσθέσεων το κάθε στοιχείο του τελικού πίνακα.

Τέλος, όσον αφορά τη main, έχουμε το iddle loop, ενώ καλούμε τις δύο συναρτήσεις για τον πολλαπλασιασμό των πινάκων.

```c
#include <avr/io.h>
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <avr/cpufunc.h>

#define N 3

volatile int int_matrix_A[3][3] = { { 1, 1, 1 },
                                    { 2, 2, 2 },
                                    { 3, 3, 3 } };
volatile int int_matrix_B[3][3] = { { 1, 1, 1 },
                                    { 2, 2, 2 },
                                    { 3, 3, 3 } };
volatile int int_matrix_C[3][3];

volatile float float_matrix_A[3][3] = { { 1.0, 1.0, 1.0 },
                                        { 2.0, 2.0, 2.0 },
                                        { 3.0, 3.0, 3.0 } };
volatile float float_matrix_B[3][3]= { { 1.0, 1.0, 1.0 },
                                       { 2.0, 2.0, 2.0 },
                                       { 3.0, 3.0, 3.0 } };
volatile float float_matrix_C[3][3];
```

(a) C code for defines

```c
int main(void){

    int_multiply(int_matrix_A, int_matrix_B, int_matrix_C);

    float_multiply(float_matrix_A,float_matrix_B, float_matrix_C);

    /*for (int i=0; i<N; i++){
        for (int j=0; j<N; j++){
            printf("%f", float_matrix_C[i][j]);
        }
    }*/
    while(1){

    }
}
```

(b) C code for main

```c
void int_multiply(volatile int A[][N], volatile int B[][N], volatile int C[][N]){
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            C[i][j] = 0;
            for (int k = 0; k < N; k++){
                C[i][j] += A[i][k] * B[k][j];
            }
        }
    }
    return;
}
```

(c) C code for INT multiplication function

```c
void float_multiply(volatile float A[][N], volatile float B[][N], volatile float C[][N]){
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            C[i][j] = 0.0;
            for (int k = 0; k < N; k++){
                C[i][j] += A[i][k] * B[k][j];
            }
        }
    }
    return;
}
```

(d) C code for FLOAT multiplication function

Προσομοίωση Αποτελεσμάτων

Όπως παρατηρούμε και στις δύο προσομοιώσεις για integers και floats, ο κάθε αριθμός που αποθηκεύουμε καταλαμβάνει χώρο στη μνήμη ίσο με 4 bytes. Συνεπώς, για την αποθήκευση ενός πίνακα 3x3 χρειαζόμαστε χώρο 36 bytes, καθώς αποθηκεύουμε 9 αριθμούς για κάθε πίνακα. Επιπλέον, βλέπουμε ότι η ρουτίνα για τον υπολογισμό του γινομένου των 2 integer πινάκων χρειάζεται 2824 - 1337 = 1487 κύκλους.

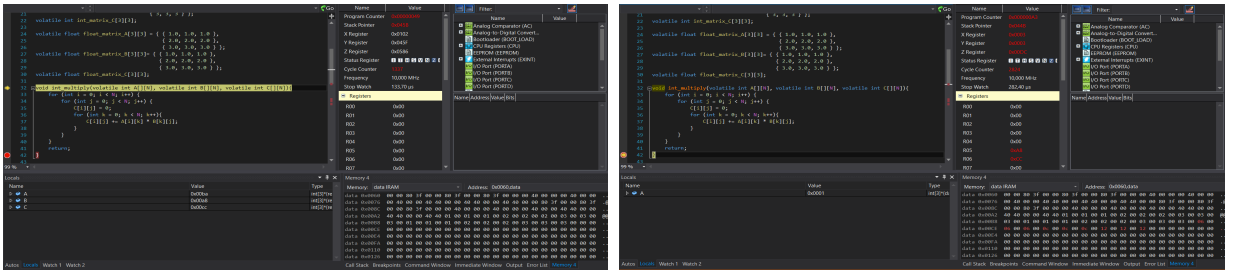

Figure 2: Results from Atmel Studio 7 - INT multiplication

Όσον αφορά τον υπολογισμό του γινομένου float αριθμών ο ίδιος αλγόριθμος όταν εντελεστεί από το μικροελεγκτή AVR, για να υπολογίσει το αποτέλεσμα χρειάζεται 11040 - 2870 = 8170 κύκλους.

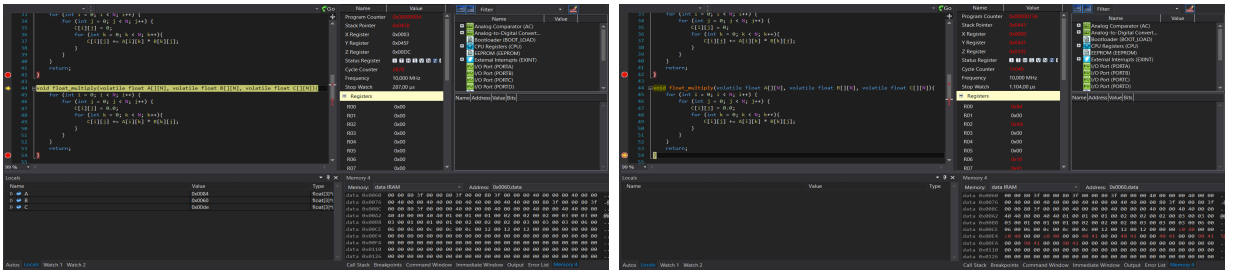

Figure 3: Results from Atmel Studio 7 - FLOAT multiplication

Όπως φαίνεται και στις παραπάνω προσομοιώσεις οι πόροι που χρησιμοποιεί το AVR είναι η μνήμη που διαβάζει τους πίνακες που πολλαπλάσιάζουμε και αποθηκεύει τα δεδομένα του αποτελέσματος, καθώς και κάποιοι καταχωρητές, για να κάνει τον πολλαπλασιασμό και το άθροισμα των αριθμών. Τέλος, παρατηρούμε ότι στον μικροελεγκτή AVR το γινόμενο integer 3x3 πινάκων χρειάζεται εμφανώς λιγότερους κύκλους σε σχέση με την περίπτωση που οι πίνακες που πολλαπλασιάζουμε είναι τύπου float.