ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

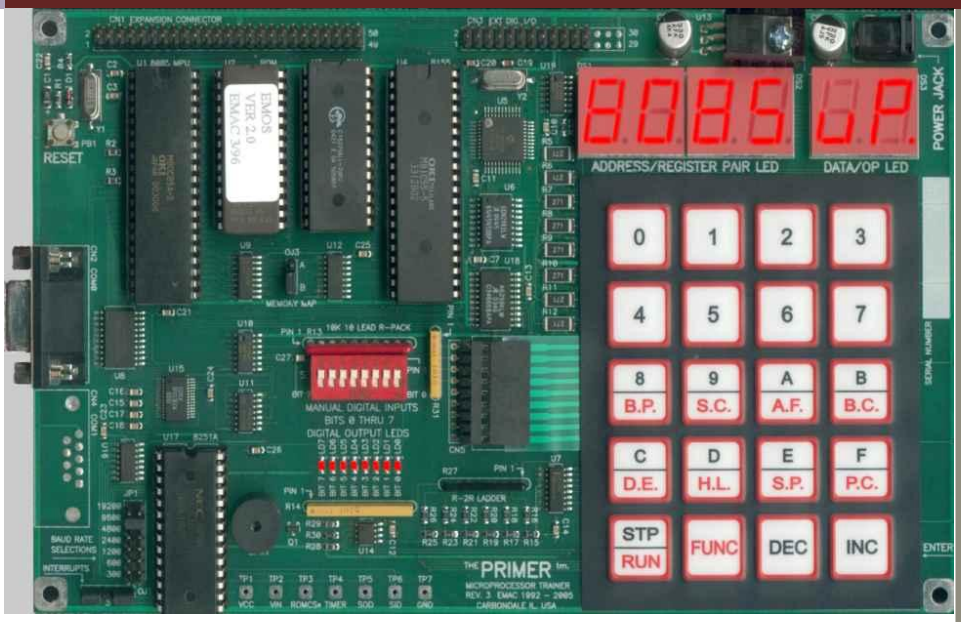
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ

&

ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Συστήματα Μικροϋπολογιστών

1η Σειρά Ασκήσεων



6ο Εξάμηνο

ΡΟΗ Υ

Αθανασίου Νικόλαος

ΑΜ 03112074

Σταυρακάκης Δημήτριος

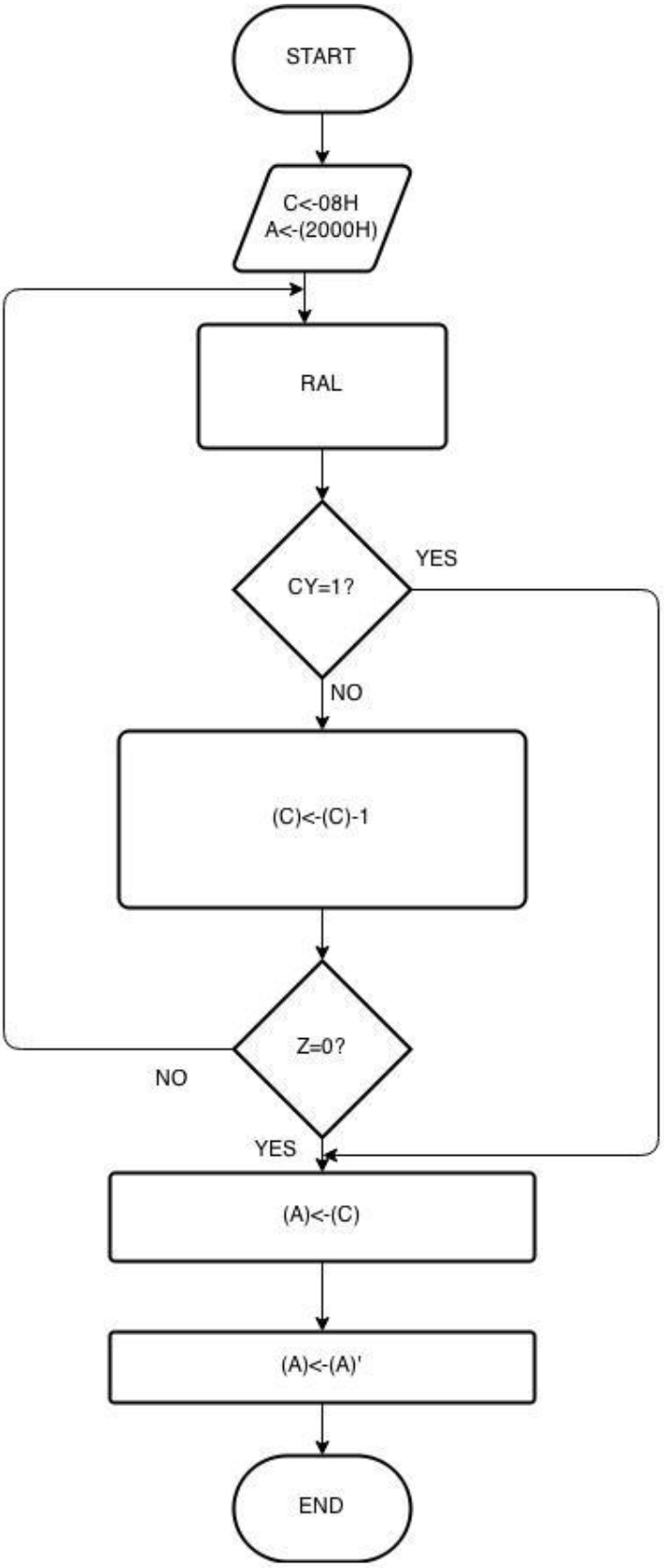
ΑΜ 03112017

ΑΣΚΗΣΗ 1

Το πρόγραμμα σε συμβολική γλώσσα αφού τη μεταφράσαμε κατάλληλα είναι:

0800	0E-opcode	***START:***	**MVI C,08H**	
0801	08-data			
0802	3A-opcode		**LDA 2000H**	
0803	00-data			
0804	20-data			
0805	17-opcode	***LOOP:***	**RAL**	
0806	DA-opcode		**JC LEDS**	(Address:080D)
0807	0D-data			
0808	08-data			
0809	0D-opcode		**DCR C**	
080A	C2-opcode		**JNZ LOOP**	(Address:0805)
080B	05-data			
080C	08-data			
080D	79-opcode	***LEDS:***	**MOV A,C**	
080E	2F-opcode		**CMA**	
080F	32-opcode		**STA 3000H**	
0810	00-data			
0811	30-data			
0812	CF-opcode		**RST 1**	

Το παραπάνω πρόγραμμα υλοποιεί το εξής:
Βρίσκει και τυπώνει τη θέση του πλέον σημαντικού άσσου που βρίσκεται στον αριθμό που εισάγεται μέσω των διακοπτών. Αφού το υπολογίσει τυπώνει στα Leds την αντίστοιχη θέση του πρώτου άσσου σε δυαδική μορφή. Πχ για είσοδο τον αριθμό 00101011 το πρόγραμμα θα τυπώσει τον αριθμό 6 που είναι η θέση του πλέον σημαντικού άσσου. Το διάγραμμα ροής που ζητείται είναι το παρακάτω:



Προκειμένου να γίνει συνεχής η ροή του προγράμματος αυτό που χρειάζεται να αλλάξουμε είναι στη θέση της εντολής RST 1 να βάλουμε την εντολή JMP START.

ΑΣΚΗΣΗ 2

```
        IN 10H          ; ACCESS IN RAM
        MVI B,01H       ;FORTWNW TON B GIA TH LEITOURGIA THS ENTOLHS DELB
        MVI C,F4H       ;FORTWNW TO C GIA TH LEITOURGIA THS ENTOLHS DELB
                        ;PARAKATW POU 8A MAS XREIASTEI
        MVI D,01H       ;DEIXNEI O D POIO LAMPAKI ANAVEI
        STC
        CMC             ; 8ETW CY=0 GIA APOFYGH PROVLHMATWN
        LDA 2000H       ;CHECKARW POIO LED THA ANAPSEI PRWTO
        RAL
        JNC FIRST_LED   ;AN EXW MSB 1 TOTE ANAVEI TO LED STH THESH 8
        MVI D,80H       ;ALLIWS ANAVEI TO PRWTO APO TA DEKSIA

FIRST_LED:  MOV A,D     ;(A)<-(D)
        CMA             ;PAIRNW SYMPLHRWMA LOGW ANASTROFHS LOGIKHS
        STA 3000H       ;TURN ON 1ST LED

START:  LDA 2000H       ;INPUT APO DIAKOPTES
        RAR             ;DE3IA OLIS8HSH:  CY= LSB
        JC FREEZE       ; AN CY=1 PAGWNW
        RAL             ;2 FORES DE3IA OLIS8HSH KAI PAIRNW TO MSB
        RAL             ; STON CY
        JNC R_L         ; AN CY=0  TOTE LEDS: DEKSIA PROS ARISTERA
        JMP L_R         ; DIAFORETIKA APO ARISTERA PROS DEKSIA

FREEZE:  MOV A,D        ;VAZW STON A TON D GIA NA PARAMEINEI H IDIA EIKONA
                        ;ME TA LEDS
        JMP TURNONLED

TURNONLED:  MOV D,A     ; (A)--> (D)
        CMA             ;PAIRNW TO SYMPLHRWMA
        STA 3000H       ;ANAVW TO LED
        CALL DELB       ;DHMIOURGW THN KA8YSTERHSH POU XREIAZOMAI
        JMP START       ;KAI KSANA APO THN ARXH

R_L:    MOV A,D         ; (A)<--(D)
        RAL             ;METAKINW TO LED MIA THESH ARISTERA
        JC RECYCLE      ; AN FTASW STO TELEYTAIO PAW PALI STO PRWTO
        JMP TURNONLED ; TO ANAVW

L_R:    MOV A,D         ; (A)<--(D)
        RRC             ; METAKINW TO LED MIA THESH DEKSIA KAI KANW STHN
                        ;OUSIA RECYCLE ME XRHSH THS RRC
        JMP TURNONLED

RECYCLE:    SUB A       ; KANW TO (A)=0
        ADI 01H         ; KANW TO (A)=1
        JMP TURNONLED
        END
```

Το παραπάνω πρόγραμμα διαβάζει κάθε φορά τον αριθμό που δίνεται από τους διακόπτες και εκτελεί τα παρακάτω:

- Αν το MSB του δοσμένου αριθμού είναι 1 τότε ανάβουν τα leds από αριστερά προς δεξιά.
- Αν το MSB του δοσμένου αριθμού είναι 0 τότε ανάβουν τα leds από δεξιά προς αριστερά.
- Αν το LSB γίνει 1 τότε εκείνη τη στιγμή «παγώνουν» τα leds και μένουν στις θέσεις όπου βρίσκονται.

Αυτό τρέχει συνεχόμενα. Έχουμε εισάγει καθυστέρηση με την εντολή DELB 0.5sec για το άναμμα των leds.

ΑΣΚΗΣΗ 3

```
START:
        LDA 2000H       ;READ INPUT
        MOV D,A         ;D CONTAINS THE STARTING VALUE OF INPUT
        MVI B,FFH       ; B = -1

DECA:
        INR B           ; B <- B+1 (THE FIRST TIME ITS CALLED B BECOMES 0)
        SUI 0AH         ; A <- A-10
        JNC DECA        ; IF A>0 JUMP TO DECA. IF NOT THEN B CONTAINS THE DECADES
        ADI 0AH         ; ADD 10 TO PROCCESS THE UNITS
        MOV C,A         ; USE C REGISTER TO SAVE THE UNITS
        MVI A,00H       ; A <-0
        ADD B           ; A <-B
        RLC             ; WE WANT THE TENS TO BE ANNOUNCED AT THE 4 MOST
        RLC             ;SIGNIFICANT BITS
        RLC             ; TO ACHIEVE THAT WE ROTATE LEFT THE NUMBER FOR FOUR TIMES
        RLC
        ADD             ; WHEN ROTATION STOPS WE ADD THE UNITS AT REG A
        CMA             ; LEDS USE COMPLEMENTERY LOGIC SO WE TAKE THE
                        ; COMPLEMENT OF A
        STA 3000H       ; LEDS "ON"
        MOV C,A         ;STORE DECADES AND UNITS IN REG C
        MOV A,D         ;RESTORE STARTING VALUE OF A
        CPI 63H         ;IS INPUT>99?
        JNC HUNDR       ;IF INPUT>99 JUMP TO HUNDR
        JMP START       ; CONTINUOUS OPERATION
HUNDR:
        MVI A,F0H       ;LIGHT 4 MSBits
        STA 3000H       ;LIGHTS UP 4 MSBits
        LXI B, 00A4
        CALL DELB
        MVI A,0FH       ;LIGHT 4 LSBits
        STA 3000H       ;LIGHTS UP THE UNITS
        LDA 2000H       ; READ INPUT AGAIN
        CMP D           ;CHECK IF IT IS DIFFERENT
        JNZ START       ;IF YOU READ NEW NUMBER CHECK AGAIN
        JMP HUNDR       ;ELSE CONTINUE WITH THE SAME PROCESS
END:
        END
```

Έχοντας ως βάση τη συνάρτηση του παραδείγματος του βιβλίου, την επεκτείνουμε ώστε να ανάβουν στα leds τα εξής:

- Αν ο αριθμός είναι κάτω του 100:
  - Στα 4 Most Significant Bits (4 πρώτα leds) τυπώνεται ο αριθμός των δεκάδων
  - Στα 4 Least Significant Bits (4 τελευταία leds) τυπώνεται ο αριθμός των μονάδων

- Αν ο αριθμός είναι από 100 και πάνω:
  - Αναβοσβήνουν εναλλασσόμενα τα 4 MSBits (4 πρώτα leds) και τα 4 LSBits (4τελευταία leds)

Αν θέλω και για αριθμούς πάνω από 100 να Αναβοσβήνουν εναλλασσόμενα ο αριθμός των δεκάδων στα 4 MSBits (4 πρώτα leds) και των μονάδων στα 4 LSBits (4τελευταία leds) τότε απλά κάνω το κομμάτι του κώδικα με label HUNDR ως εξής:

```
MOV A,C      ;A NOW CONTAINS DECADES AND UNITS
MVI E,F0H    ;CHECK 4 MSBits
ANA E
CMA
STA 3000H    ;LIGHTS UP THE DECADES
MOV A,C      ;A CONTAINS DECADES AND UNITS
MVI E,0FH    ;CHECK 4 LSBits
ANA E
CMA
STA 3000H    ;LIGHTS UP THE UNITS
LDA 2000H    ; READ INPUT AGAIN
CMP D        ;CHECK IF IT IS DIFFERENT
JNZ START    ;IF YOU READ NEW NUMBER CHECK AGAIN
JMP HUNDR    ;ELSE CONTINUE WITH THE SAME PROCESS
```

ΑΣΚΗΣΗ 4

```
START:
MVI B,00H       ;STON B KRATAME PROSWRINA T APOTELESMA THS LOGIKHS PYLHS
LDA 2000H       ;KANOUME LOAD THN EISODO APO TON A KATAXWRHTH
STC
CMC             ;MHDENISMOS TOU CY
MVI C,00H       ;MHDENISMOS TOU KATAXWRHTH C POU 8A XREIASTEI PARAKATW
MVI D,00H       ;MHDENISMOS TOU KATAXWRHTH D POU 8A XREIASTEI PARAKATW
RAR
JC OR1_1_1ST_COMPARISON     ;CHECK TOU PRWTOU BIT POU VRISKETAI STO CY KAI RAR
RAR             ;AN DEN EINAI ASOS TO PRWTO BIT KANE RAR KAI CHECKARE TO 2o
JC OR1_1        ;AN EINAI ASOS TO 2o FYGE KAI KANE TO APOTELESMA THS OR 1
JNC OR1_0       ;ALLIWS ASTO 0

OR1_1_1ST_COMPARISON:  ;SE PERIPTWSH POU VRW 1 STO PRWTO BIT KAI XWRIS NA
RAR                    ;CHECKARW TO DEYTERO APLA KANW MIA RAR KAI TON B=1


OR1_1:          ;IF OR RESULT=1
INR B           ;B=B+1=0+1=1

OR1_0:
MOV D,B         ;RESULT IN D
MVI B,00H       ;END OF 1ST OR,EPANATHESH B=0 GIA EPOMENH PYLH

RAR             ;GIA TH 2H OR AKOLOUTHEITAI IDIA DIADIKASIA ME PANW
JC OR2_1_1ST_COMPARISON
RAR
JC OR2_1
JNC OR2_0

OR2_1_1ST_COMPARISON:  ;SE PERIPTWSH POU VRW 1 STO PRWTO BIT KAI XWRIS NA
RAR                    ; CHECKARW TO DEYTERO APLA KANW MIA  RAR KAI TON  B=1

OR2_1:          ;IF OR RESULT=1
INR B

OR2_0:
MOV C,B         ;RESULT IN C,AYTH TH FORA TO VAZW STO C GIA NA KANW PARAKATW XOR
MVI B,00H       ;END OF 2ND OR,EPANATHESH B=0 GIA EPOMENH PYLH

MOV E,A         ;KRATAW TO PERIEXOMENO TOU A
MOV A,C         ;VAZW TON C STON A
CMP D           ;CHECKARW XOR,SYGKRINONTAS ME D POU EXEI TO APOTELESMA 1HS OR
STC
CMC             ;MHDENIZW CY
RAL             ;OLIS8AINW GIA NA PARW TO 2o BIT POU 8ELW POU VRISKETAI ;STHN 1H
                ;THESH TOY C
MOV C,A         ;PERNAW TON A STON C KAI EXW ETOIMO TO 2o BIT
JZ SKIP         ;AN EXW 0 STHN XOR PAW STO TAG SKIP (Z=1 APO CMP D) ALLIWS
INR C           ;AFOU EINAI LSB H XOR ME +1 ENHMERWNW AN H XOR EINAI 1
```

```
SKIP:
MOV A,E         ;PERNAW TON E STON A
RAR             ;ELEGXOS PRWTOU BIT GIA AND1
JNC AND1_FIRST_COMPARISON          ;AN 1o BIT EINAI MHDEN FYGE
RAR
JNC AND1        ;ALLIWS CHECKAROUME KAI TO EPOMENO AN EINAI MHDEN
INR B           ;AN EINAI KAI OI 2 ASSOI APOTELESMA STON B=1
JC AND1

AND1_FIRST_COMPARISON:
RAR             ;AN HDH TO PRWTO STOIXEIO EINAI 0 APLA OLIS8HSE MIA 8ESH

AND1:
MOV E,A         ;PERNAW TON A STON E
MOV A,B         ;PERNAW TON B STON A
STC
CMC             ;MHDENIZW TON CY
RAL
RAL             ;KANW DYO OLIS8HSEIS ARISTERA GIA NA PAW T BIT STH SWSTH 8ESH
ADD C           ;PROS8ETW A+C, K EXW 3 PRWTA BIT SWSTA STON A
MOV C,A         ;METAFERW TON A STO C
MOV A,E         ;EPANAFORA TIMHS A KAI END 1ST AND
MVI B,00H       ;START 2ND AND, KANW TON B PALI 0
RAR             ;AKOLOU8W IDIA DIADIKASIA ME THN PARAPANW
JNC AND2_FIRST_COMPARISON
RAR
JNC AND2
INR B
JC AND2

AND2_FIRST_COMPARISON:
RAR

AND2:
MOV E,A
MOV A,B
STC
CMC
RAL
RAL
RAL             ;3 FORES OLISTHISI EDW GIA NA ETOIMASOUME TO 4o BIT
ADD C           ;PROS8ETOUME TON A STON C KAI EXOUME ETOIMA KAI TA 4 BIT
MOV C,A         ;END 2ND AND
RESULTS_IN_LEDS:
MOV A,C         ;METAFEROUME TO APOTELESMA MAS STON A
CMA             ;EPEIDH EXOUME LEDS ARNHTIKHS LOGIKHS PAIRNOUME SYMPLHRWMA
STA 3000H       ;TO TOPO8ETOUME STH DIEY8YNSH 3000H GIA NA DOUME TA LEDS
RST
END
```

Ο κώδικας που παρατίθεται παραπάνω ελέγχει bit προς bit την είσοδο που του δίνουμε με τους διακόπτες και υλοποιεί το σύστημα λογικών πυλών που δίνεται. Πιο συγκεκριμένα αν βρει μια εκ των δύο εισόδων της OR να ισούται με ένα κάνει αμέσως το αποτέλεσμα της OR ίσο με 1. Αντίστοιχα αν βρει σε AND πύλη bit εισόδου 0, κάνει το αποτέλεσμα της AND άμεσα 0. Τσεκάρω μετά και το αποτέλεσμα της XOR σύμφωνα με τις τιμές αληθείας της ανάλογα με τα bit εισόδου. Τα αποτελέσματα των πυλών αυτών αποθηκεύονται στους καταχωρητές και με ολισθήσεις παίρνουν την κατάλληλη θέση σημαντικότητάς τους.

ΑΣΚΗΣΗ 5

Σύμφωνα με τα δεδομένα της άσκησης, η συνάρτηση του κόστους για κάθε τεχνολογία είναι:

Διακριτά Στοιχεία: $K_{\Delta\Sigma}(x) = 20000 + (10+10)x = 20000 + 20x$

FPGAs: $K_{FPGAs}(x) = 10000 + (30+10)x = 10000 + 40x$

SoC-1: $K_{SoC-1}(x) = M + (2+2)x = M + 4x$

SoC-2: $K_{SoC-2}(x) = 200000 + (1+1)x = 200000 + 2x$

Αντίστοιχα, οι συναρτήσεις κόστους ανα τεμάχιο υπολογίζονται:

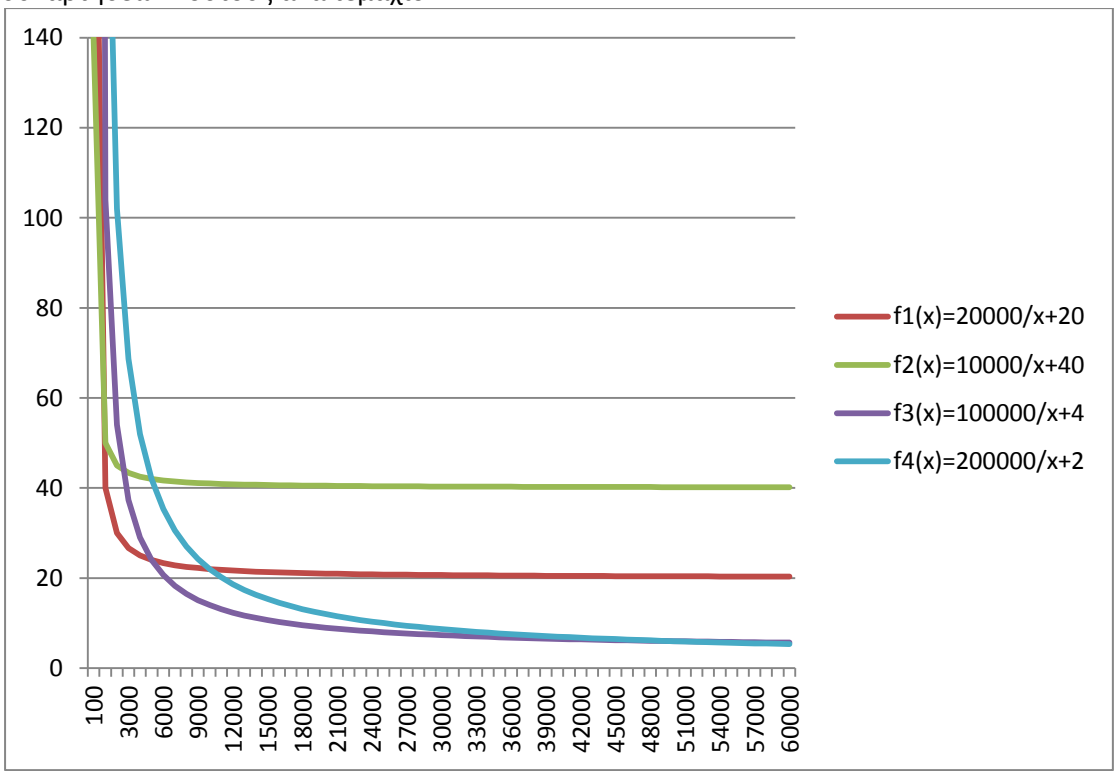
Διακριτά Στοιχεία: $K_{\Delta\Sigma}(x) = \frac{20000}{x} + (10+10) = \frac{20000}{x} + 20$

FPGAs: $K_{FPGAs}(x) = \frac{10000}{x} + (30+10) = \frac{10000}{x} + 40$

SoC-1: $K_{\Delta\Sigma}(x) = \frac{100000}{x} + (2+2) = \frac{100000}{x} + 4$

SoC-2: $K_{\Delta\Sigma}(x) = \frac{200000}{x} + (1+1) = \frac{200000}{x} + 2$

Στη συνέχεια παρατίθενται σε κοινό διάγραμμα οι γραφικές παραστάσεις των συναρτήσεων κόστους ανά τεμάχιο:

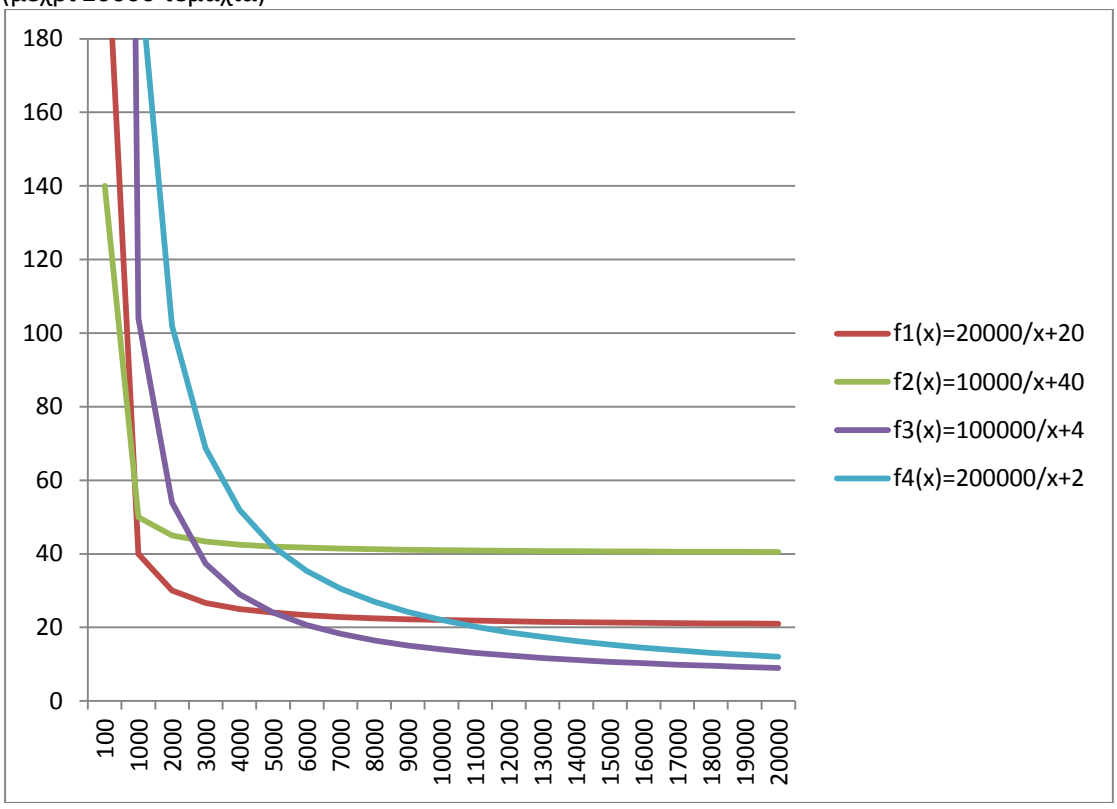


Τα χρώματα αντιστοιχούν σε:

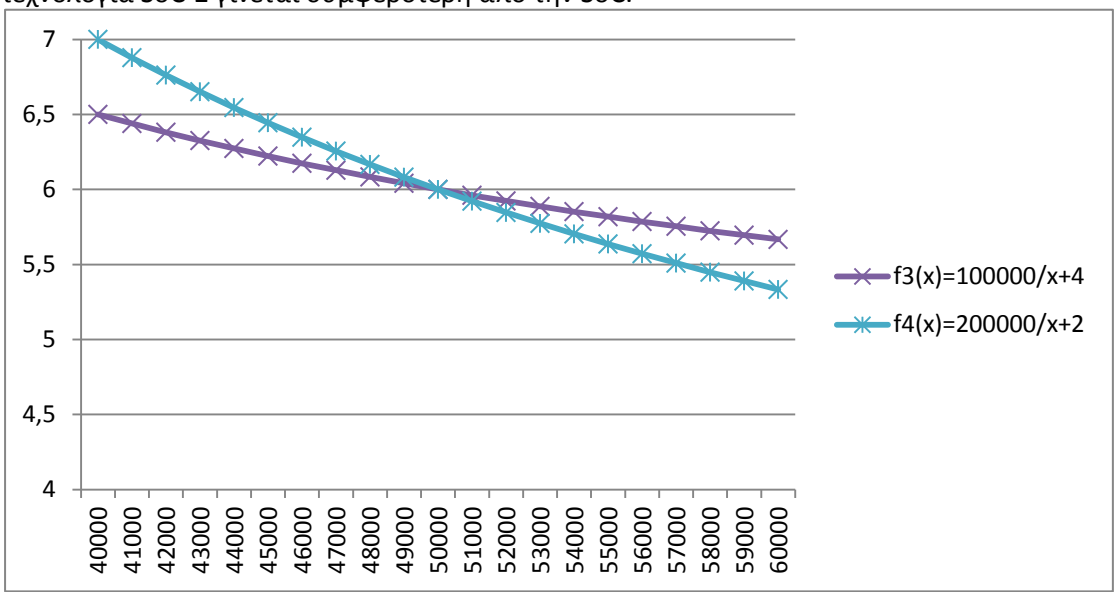
- Μπορντώ: Διακριτά στοιχεία
- Πράσινο: FPGAs
- Μωβ: SoC-1
- Μπλε: SoC-2

Ο οριζόντιος άξονας εκφράζει τον αριθμό τεμαχίων
και ο κατακόρυφος άξονας δείχνει το κόστος τεχνολογίας ανά τεμάχιο.

Για να γίνει πιο ξεκάθαρη η διαφορά τους παρατίθενται τα ακόλουθα δύο διαγράμματα:
(μέχρι 20000 τεμάχια)



Και παρακάτω παρουσιάζεται το διάστημα όπου γίνεται φανερό το σημείο όπου η τεχνολογία SoC-2 γίνεται συμφερότερη από την SoC.



- Από τα παραπάνω διαγράμματα παρατηρούμε ότι μέχρι τα 500 περίπου τεμάχια υπερτερεί η τεχνολογία των FPGAs.
- Από 500 μέχρι 5000 περίπου τεμάχια υπερτερεί όλων η τεχνολογία των διακριτών στοιχείων.
- Από 5000 μέχρι 50000 περίπου τεμάχια η πιο συμφέρουσα είναι η τεχνολογία SoC-1.
- Από 50000 τεμάχια και πάνω χαμηλότερο κόστος ανά τεμάχιο έχει η τεχνολογία SoC-2, παρά το γεγονός ότι το αρχικό κόστος σχεδίασης είναι πολύ μεγαλύτερο από τις προηγούμενες κατηγορίες.
  Επομένως για μεγάλες παραγωγές τεμαχίων συμφέρει περισσότερη η SoC-2.

Στη συνέχεια, θεωρούμε ως άγνωστο (N) την τιμή των I.C. στη τεχνολογία των FPGAs. Η συνάρτηση κόστους γίνεται:

$$\mathrm{K}_F(x) = 10000 + (N+10)\cdot x$$

Αντίστοιχα θεωρούμε ως άγνωστο (M) την τιμή αρχικού κόστος SoC-1 οπότε η συνάρτηση κόστους γίνεται:

$$K_{SoC-1}(x) = M + (2+2)x = M + 4x$$

Για να εξαφανιστεί η επιλογή της τεχνολογίας των διακριτών στοιχείων θα πρέπει να ισχύει:

$$\mathrm{K}_{\Delta\Sigma}(x) - \mathrm{K}_F(x) > 0 \Leftrightarrow 20000 + 20x - 10000 - Nx - 10x > 0$$

$$\mathrm{K}_{\Delta\Sigma}(x) - \mathrm{K}_{SoC-1}(x) > 0 \Leftrightarrow 20000 + 20x - M - 4x > 0$$

Όπως βλέπουμε οι επιτρεπόμενες τιμές των N,M εξαρτώνται από το πλήθος τεμαχίων που παράγουμε. Επιλύοντας τις παραπάνω ανισώσεις έχουμε:

$N < \frac{10000}{x} + 10$ Αυτές οι ανισώσεις δίνουν για κάθε αριθμό τεμαχίων τις οριακές τιμές.

$M < 20000 + 16x$

Επομένως για κάποια τυχαία τιμή, έστω x=1000, έχω: N<20€ και M<36000€.
Για x=20000, έχω: N<10.5€ και M<340000€.Επομένως παρατηρούμε ότι ανάλογα με την παραγωγή που θέλουμε μπορούμε να μεταβάλλουμε κατάλληλα τα N,M ώστε οι τεχνολογίες των FPGAs και SoC να μπορούν να εξαφανίσουν την τεχνολογία των διακριτών στοιχείων.