Εργαστηριακή άσκηση 3

Τομαράς Παναγιώτης-Δημήτριος en1200002

Σύστημα δειγματοληψίας δεδομένων και καταχώρησης σε μνήμη τύπου block ram.

Περιεχόμενα

[Εισαγωγή 2](#_Toc56709832)

[Σχεδιασμός – αναλυτική περιγραφή μονάδων και λειτουργιών 3](#_Toc56709833)

[Sync\_stage 3](#_Toc56709834)

[Median\_filter 3](#_Toc56709835)

[Fsm\_wr 3](#_Toc56709836)

[Picoblaze 4](#_Toc56709837)

[Σχεδιαστική επιλογή στον Picoblaze 5](#_Toc56709838)

[Interface 5](#_Toc56709839)

[Χρονισμός 5](#_Toc56709840)

[Έλεγχος σωστής λειτουργίας 6](#_Toc56709841)

[Sync\_stage 6](#_Toc56709842)

[Median\_fliter 6](#_Toc56709843)

[Fsm\_wr 7](#_Toc56709844)

[Σχηματική αναπαράσταση κυκλώματος 8](#_Toc56709845)

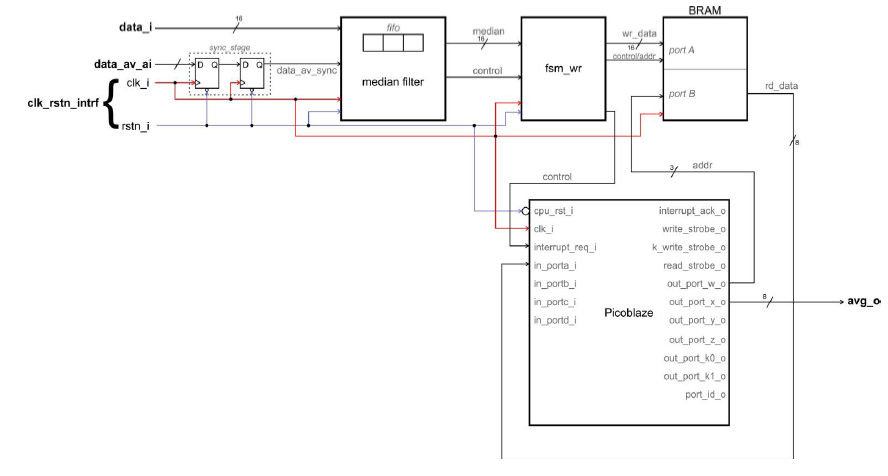
# Εισαγωγή

Στην άσκηση αυτή υλοποιούμε ένα σύστημα δειγματοληψίας δεδομένων και καταχώρησης σε μνήμη τύπου block RAM.

Τα δεδομένα 16-bit παράγονται και δειγματοληπτούνται με ασύγχρονο τρόπο και επεξεργάζονται πριν την καταχώρηση τους από ένα φίλτρο τύπου median μήκους 3 θέσεων. Τα δεδομένα ολισθαίνουν σειριακά σε μία μνήμη FIFO 3 θέσεων και σε κάθε νέα εισαγωγή το φίλτρο υπολογίζει την median τιμή και την προωθεί για καταχώρησης τη μνήμη.

Στην συνέχεια οι τιμές καταχωρούνται, με τη χρήση μηχανής καταστάσεων σε μία μνήμη Block RAM 8x16-bit και η διαδικασία δειγματοληψίας σταματάει όταν γεμίσουν και οι 8 θέσεις της μνήμης.

Όταν γεμίσει η μνήμη, με τη βοήθεια του μικροελεγκτή picoblaze διαβάζουμε τις 8 αυτές τιμές και υπολογίζουμε το μέσο όρο τους. Με το Picoblaze μειώνουμε το πλήθος και την πολυπλοκότητα των μηχανών καταστάσεων για τον έλεγχο του συστήματος.



# Σχεδιασμός – αναλυτική περιγραφή μονάδων και λειτουργιών

## Sync\_stage

Επειδή τα δεδομένα δειγματοληπτούνται με ασύγχρονο τρόπο θέλουμε ένα αρχικό στάδιο sync stage. Στο στάδιο αυτό έχουμε δύο D flip flop δηλαδή ένα Pipeline δύο σταδίων για να αντιμετωπίσουμε τη μετασταθή κατάσταση, η οποία θέλει 2 κύκλους για κάθε είσοδο. Με τον τρόπο αυτό παίρνουμε στο σύστημά μας ένα σύγχρονο σήμα.

## Median\_filter

Έπειτα έχουμε τη μονάδα median filter. Η λειτουργία αυτής της μονάδας είναι να υπολογίζει το median αριθμό μίας FIFO.

Για να υλοποιήσω αυτή τη μονάδα επέλεξα να δημιουργήσω έναν πίνακα 3 θέσεων, ένα δείκτη στη θέση πίνακα που πρέπει να γράψω ανά χρονική στιγμή, χωρίς να κάνω ολίσθηση.

Έχουμε 3 processes που τρέχουν. Τα 2 είναι σύμφωνα με το ρολόι και το άλλο είναι συνδυαστικό. Στο πρώτο Process υλοποιώ την αναμονή για το σήμα ενεργοποίησης data\_av\_sync, το οποίο πυροδοτεί την καταχώρηση στον πίνακα του αριθμού που παίρνω ως είσοδο και την αύξηση του δείκτη θέσης του πίνακα.

Ταυτόχρονα, το 2ο process παρακολουθεί πότε θα πάρω σήμα ενεργοποίησης και τη στιγμή εκείνη βγάζει στην έξοδο median\_en=1 δηλώνοντας πως η τιμή που βρίσκεται στην έξοδο median είναι αποδεκτή. Αυτό βασίζεται στη λειτουργία του 3ου process το οποίο είναι συνδυαστικό και πυροδοτείται από κάθε αλλαγή που πραγματοποιείται στο array που κρατάω τις τιμές. Ο τρόπος που υπολογίζει το median αριθμό είναι με επαναλαμβανόμενα if, όπου αρχικά βρίσκει το Index του μικρότερου αριθμού από τους 3, και ύστερα συγκρίνει τους 2 εναπομείναντες αριθμούς κρατώντας το μικρότερο, ο οποίος είναι και ο μεσαίος.

## Fsm\_wr

Επόμενο stage στη μονάδα είναι το fsm\_wr. Στο στάδιο αυτό θέλουμε να λαμβάνουμε έγκυρες τιμές median αριθμών και να τις καταχωρούμε στη block ram. Για το σκοπό αυτό έχουμε τη λειτουργία μιας fsm μηχανής 3 καταστάσεων. Οι καταστάσεις αυτές είναι οι IDLE, WRITE, PREPARE.

Στην πρώτη έχουμε το σημείο ηρεμίας της μηχανής, όπου οι έξοδοι write enable=0, read enable=0 εφόσον δε γράφουμε ούτε ενεργοποιούμε την fsm\_rd. Αναμένουμε το σήμα median\_en, δηλαδή έναν έγκυρο αριθμό ώστε να προχωρήσουμε στο στάδιο WRITE για να το γράψουμε στη μνήμη. Αμέσως βγάζουμε έξω την τιμή βασιζόμενοι στο ότι η μνήμη γράφει μόνο όταν παίρνει write enable σήμα, το οποίο ενεργοποιούμε στον επόμενο κύκλο, κι έτσι αποφεύγουμε τη χρήση ενδιάμεσου καταχωρητή για την αποθήκευση της τιμής του median.

Στην κατάσταση WRITE ενεργοποιούμε το σήμα write enable και στην έξοδο address to write βγάζουμε τη διεύθυνση μνήμης που θέλουμε να γραφτεί η τιμή.

Στο 3ο και τελευταίο στάδιο, PREPARE, αυξάνουμε το μετρητή της διεύθυνσης για το γράψιμο της μνήμης και ελέγχουμε αν έχουμε φτάσει στις 8 εγγραφές, γεγονός που μας κάνει να ενεργοποιήσουμε με σήμα read enable το μικροελεγκτή Picoblaze ώστε να αρχίσει το διάβασμα των 8 αριθμών που έχουν αποθηκευτεί και να υπολογίσει το μέσο όρο τους.

## Picoblaze

O μικροελεγκτής αναλαμβάνει τη δουλειά της fsm\_rd που συναντήσαμε στην προηγούμενη άσκηση.

Είναι μικροελεγκτής των 8 bit και προγραμματίζεται σε γλώσσα assembly. Διαθέτει μια θύρα εισόδου in\_port και μια θύρα εξόδου out\_port. Εφόσον θέλουμε να βγάζουμε δεδομένα σε 4 διαφορετικές πόρτες αλλά και να λαμβάνουμε σε 4 πόρτες, απαιτείται να προσθέσουμε ένα επιπλέον επίπεδο λογικής γύρω του.

Για να υλοποιήσουμε αυτήν την απαίτηση αξιοποιούμε τα δεδομένα από την port\_id, η οποία δηλώνει τη θύρα προορισμού, έναν πολυπλέκτη 4 σε 1 και έναν αποπλέκτη 1 σε 4. Με τις εντολές INPUT, OUTPUT, OUTPUTK η έξοδος port\_id φανερώνει τον αριθμό της πόρτας από την οποία περιμένουμε δεδομένα. Με κωδικοποίηση one-hot της port\_id και του σήματος write\_strobe αναθέτουμε την έξοδο Out\_port του μικροελεγκτή στην επιθυμητή πόρτα από αυτές που δημιουργούμε γύρω του (κωδικοποίηση one-hot 8 bit, άρα 8 πόρτες). Αντίστοιχα, στην είσοδο έχουμε έναν πολυπλέκτη που επιλέγει την πόρτα εισόδου ανάλογα με το port\_id και την προωθεί στην είσοδο in\_port του Picoblaze.

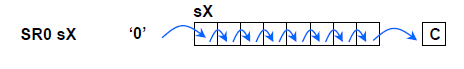
Για το άθροισμα των 8 αριθμών δε μας φτάνει ένας καταχωρητής των 8 bit. Επομένως, αξιοποιούμε 2 καταχωρητές ως εξής :

* ADD s7,s1
* ADDCY s8, 00

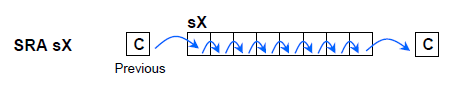
Ο s7 και ο s8 αξιοποιούνται για το άθροισμα. Με την εντολή ADD προσθέτουμε τον αριθμό που διαβάσαμε από τη μνήμη και αποθηκεύσαμε στον s1, στον s8. Αυτή η πράξη μπορεί να δημιουργήσει υπερχείλιση οπότε το flag carry γίνεται 1. Με την εντολή ADDCY προσθέτουμε στον s8 το ουδέτερο στοιχείο της πρόσθεσης με τη διαφορά ότι προστίθεται ΚΑΙ το κρατούμενο από την προηγούμενη πρόσθεση.

Για το μέσο όρο των 8 αριθμών εκτελούμε:

* SR0 s8
* SRA s7

Η δεξιά ολίσθηση στον s8 μπορεί να δημιουργήσει κρατούμενο 

Επομένως, αν έχουμε κρατούμενο από την πρώτη ολίσθηση, τότε αυτό με την εντολή SRA στον s7 θα καταλάβει τη MSB θέση.



### Σχεδιαστική επιλογή στον Picoblaze

Με τη σκέψη ότι αυτό το σύστημα μπορεί να αποτελέσει μέρος ενός μεγαλύτερου συστήματος σκέφτηκα και υλοποίησα την έξοδο ενός επιπλέον σήματος avg\_o\_en, enable το οποίο είναι active high τη στιγμή που εξάγεται ο μέσος όρος. Για την υλοποίηση αυτή πρόσθεσα στον picoblaze τις εντολές:

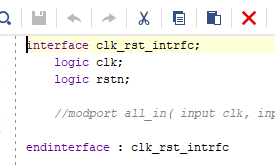
* OUTPUTK 01,out\_portk0
* OUTPUTK 00,out\_portk0

Με τις εντολές αυτές ενεργοποιώ το σήμα enable μετά τον υπολογισμό του μέσου όρου και αμέσως μετά κατεβάζω το σήμα enable.

Στο wrap επίπεδο πρόσθεσα την εντολή: «assign avg\_o\_en=out\_portk0\_o[0]» κρατώντας μόνο το LSB από την πόρτα εξόδου εφόσον το σήμα enable είναι 1 bit.

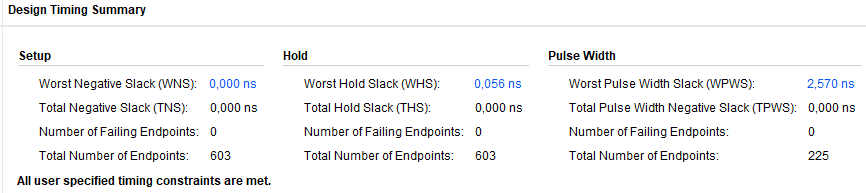
## Interface

Σύμφωνα με την παρακάτω εικόνα έχω υλοποιήσει το interface επικοινωνίας ανάμεσα στα διαφορετικά κομμάτια του συστήματός μας.



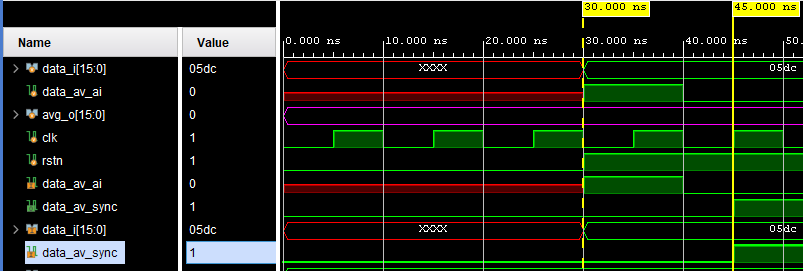
# Χρονισμός

Με το αρχείο δήλωσης constraints έχουμε ρυθμισμένη την περίοδο σε 10ns (100MHz). Με τη διαδικασία της synthesis στο Timing report αποτυπώθηκε πως το Worste Negative Slack είναι θετικό, οπότε έχουμε περιθώριο για μείωση της περιόδου, άρα αύξηση της συχνότητας του ρολογιού. Επομένως, μειώσαμε την περίοδο σε 7.641ns και το WNS σε 0 ns. Η συχνότητα λειτουργίας, πλέον, είναι 130.9 MHz.



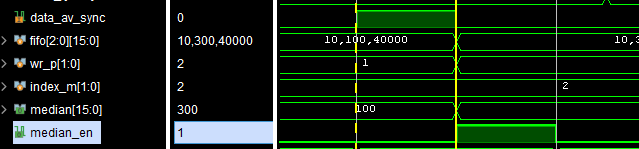
# Έλεγχος σωστής λειτουργίας

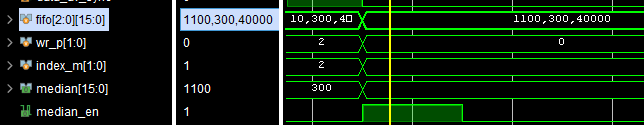
## Sync\_stage



Βλέπουμε τη λειτουργία του sync stage όπου σε 2 κύκλους ενεργοποιείται το σήμα data av sync έπειτα από την ενεργοποίηση του data\_av\_ai

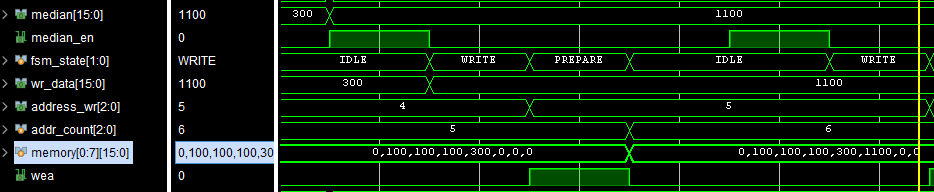
## Median\_fliter





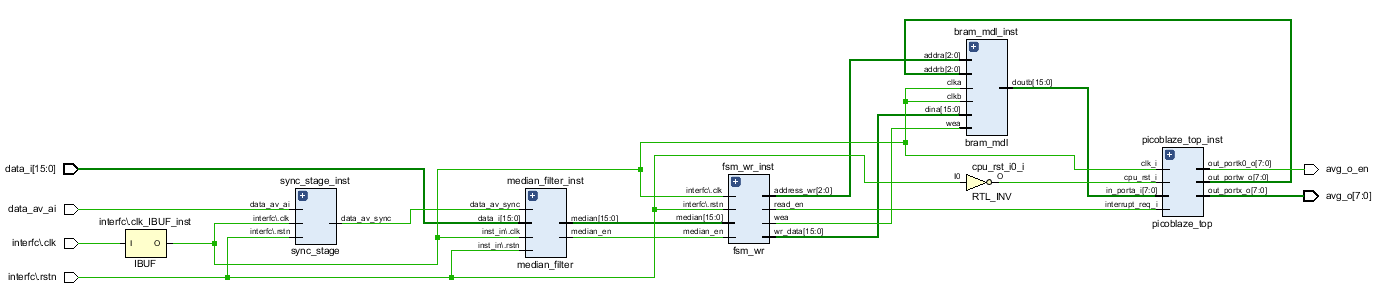
Εδώ βλέπουμε τη λειτουργία του median φίλτρου. Παίρνει σήματα ενεργοποίησης data\_av\_sync και στον επόμενο κύκλο βγάζει σήμα median\_en με την median=300 ως έγκυρη τιμή όταν η FIFO είναι η 10,300,40000. Αντίστοιχα βλέπουμε πως όταν έχουμε 1100,300,40000 ο μέσος είναι ο 1100.

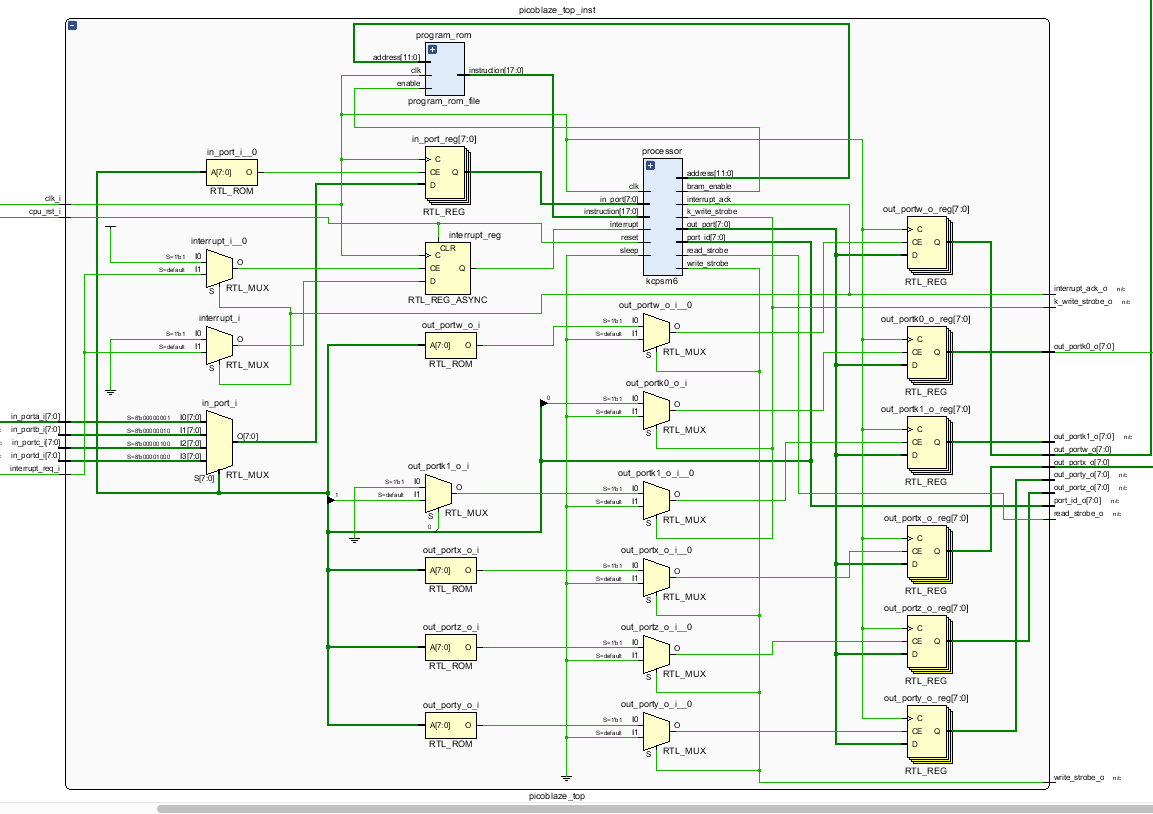
## Fsm\_wr



Εδώ έχουμε την fsm\_wr. Λαμβάνει median=1000 και σήμα ενεργοποίησης median\_enable. Αυτό βγάζει αμέσως στο wr\_data την τιμή 1000, στον επόμενο κύκλο δηλαδή στο WRITE ενεργοποιεί το write enable και δίνει τη διεύθυνση address\_wr = 4 για να γραφτεί στην 5η θέση. Στο PREPARE απλώς αυξάνω το μετρητή για τη διεύθυνση εγγραφής για την επόμενη εγγραφή. Βλέπουμε πως η μνήμη στον επόμενο κύκλο ύστερα από την παροχή write enable γράφει τον πίνακα που κρατάει τις τιμές, αλλάζοντας σε 0 100 100 100 300 1100 0 0 από αρχικά 0 100 100 100 300 0 0 0.

# Σχηματική αναπαράσταση κυκλώματος





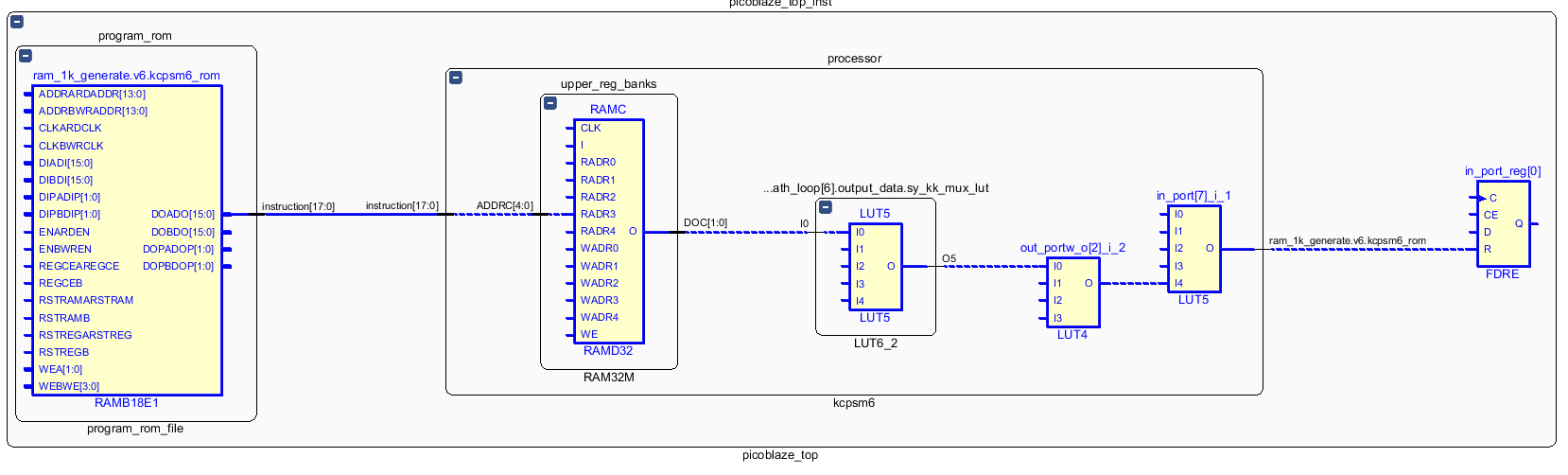
Figure 1 Σχηματικό του Picoblaze. Διακρίνονται ο πολυπλέκτης στην είσοδο που οδηγεί στον καταχωρητή για το in\_port και η λογική αποπολυπλεξίας με τα flip-flop για τις πόρτες εξόδου

Figure 2 Το πιο αργό μονοπάτι του συστήματος