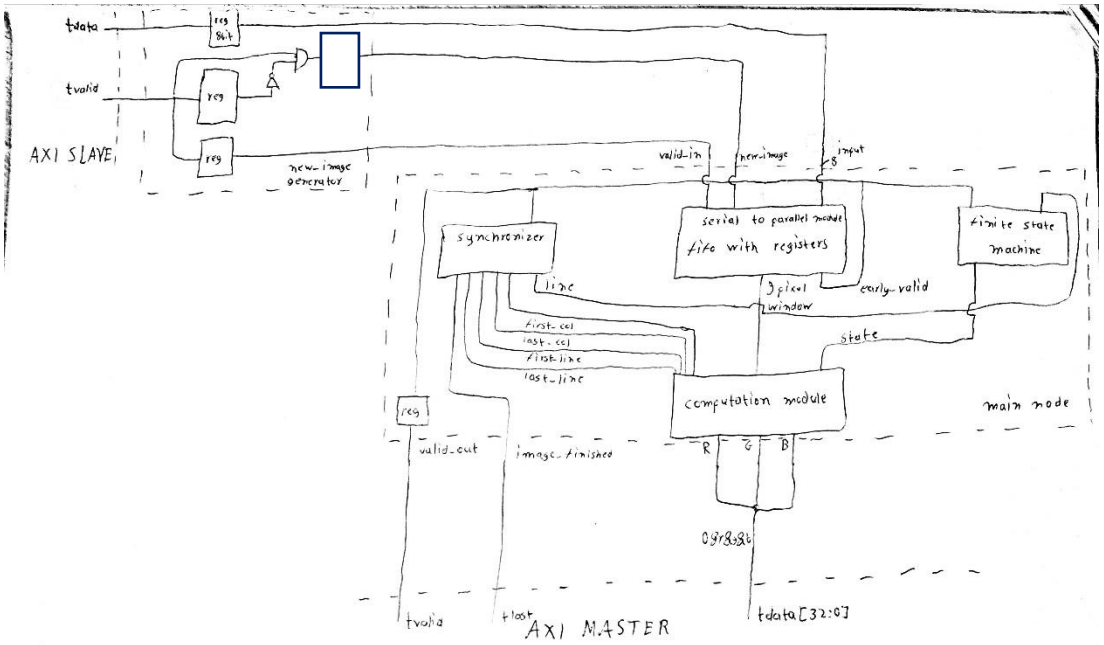
5η Εργαστηριακή Άσκηση

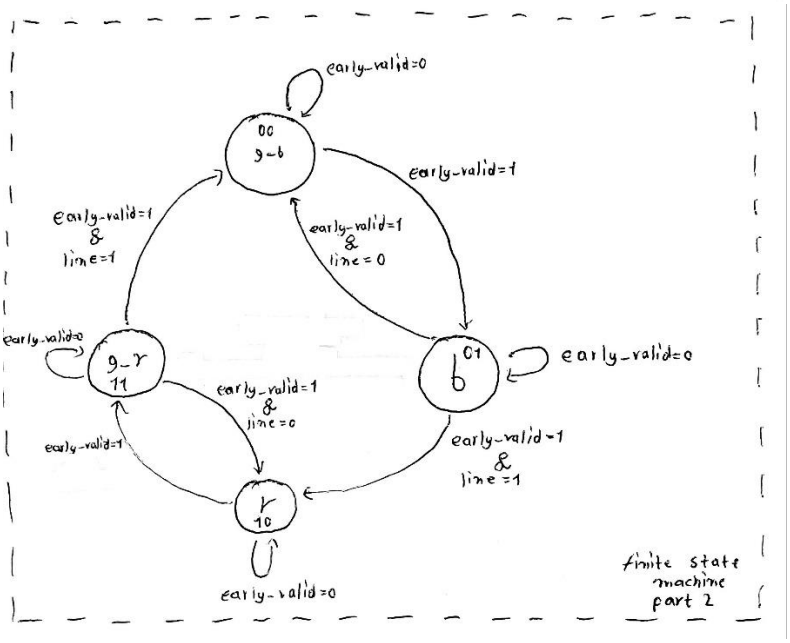
VLSI

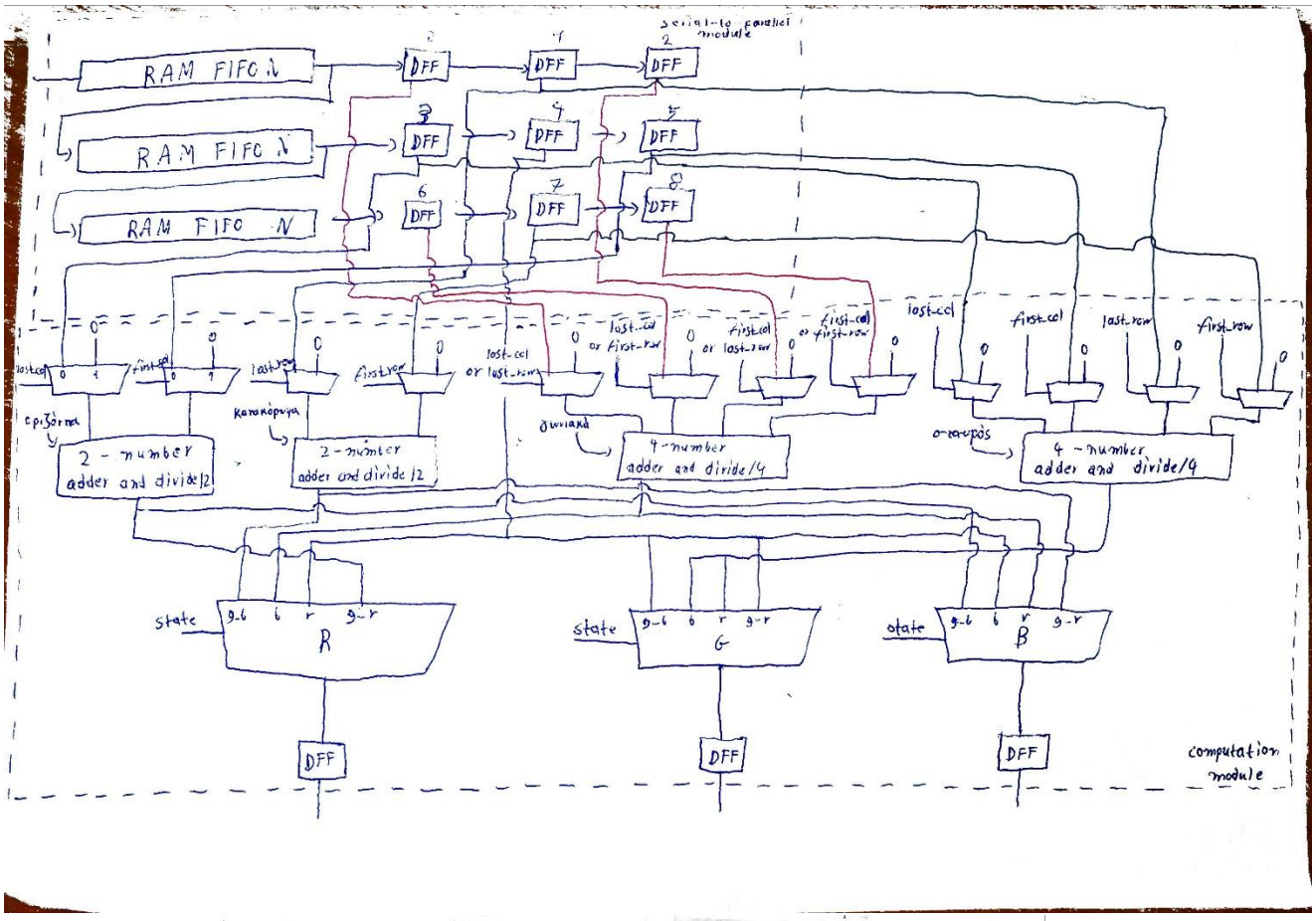
Μοίρας Αλέξανδρος 03118081

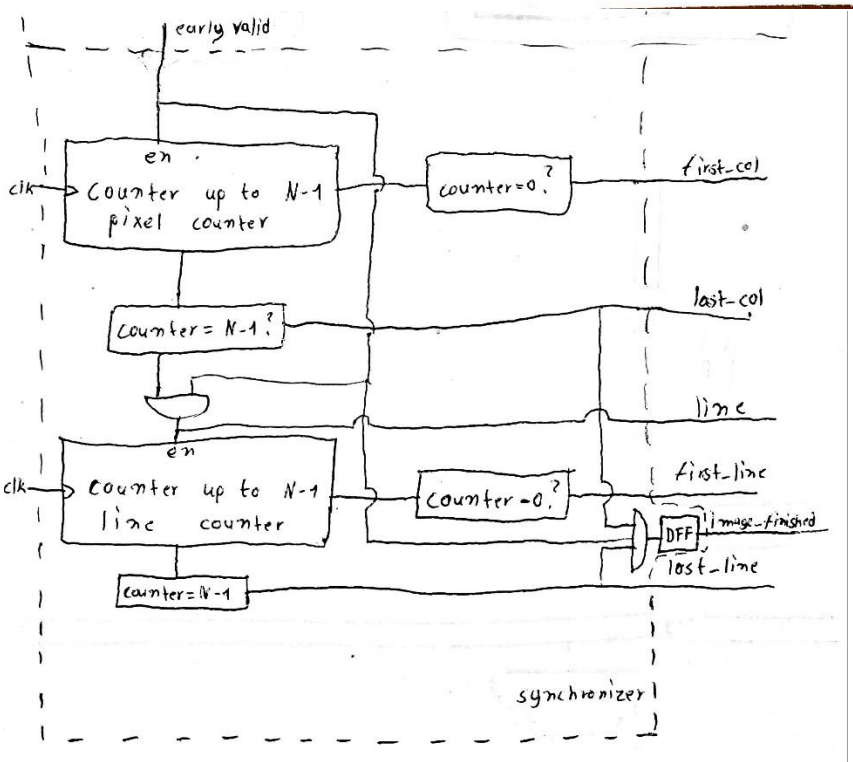
Παντελαίος Δημήτριος 03118049

Το block διάγραμμα του κυκλώματος και το finite state machine φαίνονται παρακάτω:






RAM FIFO R
RAM FIFO G
RAM FIFO B
DFF
serial-to-parallel module
last_col
first_col
last_row
first_row
last_col or first_row
first_col or last_row
2 - number adder and divide/2
2 - number adder and divide/2
4 - number adder and divide/4
4 - number adder and divide/4
state
R
G
B
computation module


early valid
en
clk
Counter up to N-1
pixel counter
counter=0 ?
first_col
counter = N-1 ?
last_col
line
Counter up to N-1
line counter
counter=0 ?
first_line
DFF
image_finished
last_line
counter=N-1
synchronizer

Το new_image_generator δημιουργεί το σήμα new_image στο πρώτο valid_in που θα λάβει από το AXI Slave Interface. Αν και στον τρέχοντα σχεδιασμό μας το new_image είναι redundant το συμπεριλαμβάνουμε για λόγους πληρότητας. Προστίθενται registers στα valid_in, pixel, new_image για συγχρονισμό τους με το ρολόι.

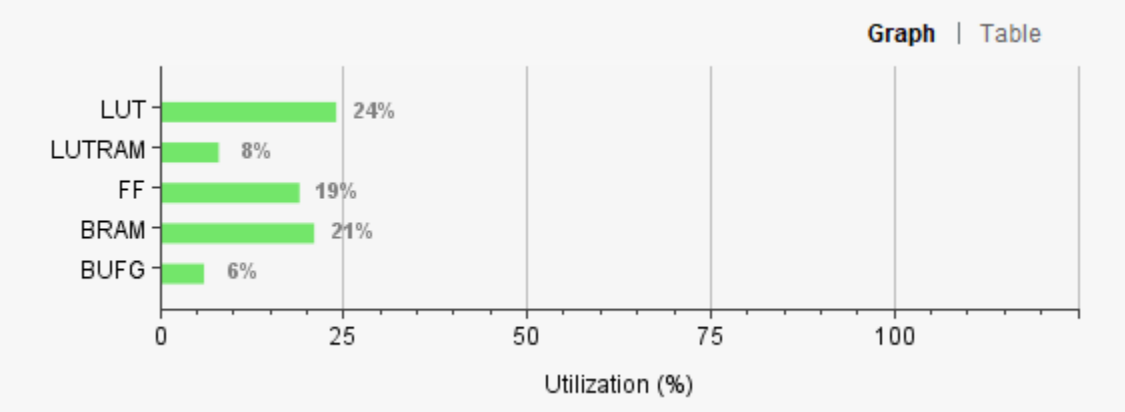
Το serial to parallel module αναλαμβάνει να δημιουργήσει το 3x3 παράθυρο. Το valid_in αποτελεί το wr_enable της πρώτης FIFO ενώ διαθέτει εσωτερικά μετρητές για τον συγχρονισμό των υπόλοιπων wr_enable και rd_enable. Όσο δεν έχουν διαβαστεί ακόμα όλα τα δεδομένα αν σταματήσει να έρχεται valid_in=1 οι FIFO σταματούν να διαβάζονται και να γράφονται και οι registers να shiftάρουν δεδομένα έως ότου έρθει νέο valid_in=1 οπότε και συνεχίζεται η κανονική λειτουργία του κυκλώματος. Επίσης όταν έχουν διαβαστεί δύο συνεχόμενα δεδομένα από τη $2^{η}$ FIFO, δηλαδή το pixel στη μέση του παραθύρου είναι valid τίθεται στο 1 το σήμα early_valid, που σηματοδοτεί ότι μπορούν να εκτελεστούν υπολογισμοί για αυτό το Pixel.

Ο synchronizer μετράει τα επεξεργαζόμενα pixels. Ο ρόλος του είναι να καθορίσει αν βρισκόμαστε στα άκρα της εικόνας ώστε να λάβει σωστά μηδενικές τιμές για τα αντίστοιχα pixel το κύκλωμα που εκτελεί τους υπολογισμούς και να σηματοδοτήσει το τέλος της επεξεργασίας όταν θα έχουν επεξεργαστεί επιτυχώς N^2 pixels. Επίσης παράγει σήμα κάθε φορά που θα μετρήσει N pixels για να υποδείξει αλλαγή γραμμής στην εικόνα.

Το FSM καθορίζει τι pixel επεξεργαζόμαστε ώστε το module των υπολογισμών να αναγνωρίζει σωστά τα χρώματα της 3x3 γειτονιάς. Αλλάζει κατάσταση για κάθε έγκυρο pixel που επεξεργάζεται καθώς και επίσης μετά την επεξεργασία N pixels όπου δέχεται το σήμα line από τον synchronizer για να πάει στα pixels της επόμενης γραμμής της εικόνας.

Το computation module δέχεται το 3x3 παράθυρο, πρώτα ελέγχεται αν κάποιο pixel πρέπει να θεωρηθεί 0 βάσει των σημάτων των synchronizer, ύστερα υπολογίζει τους 4 δυνατούς μέσους (γραμμή, στήλη, σταυρός, γωνιακά βλ. σχήμα 3 εκφώνησης) και τέλος βάσει του state τα RGB παίρνουν τιμές είτε από τους παραπάνω υπολογισμούς είτε απευθείας από το μεσαίο pixel του παραθύρου βάσει του state που υποδεικνύει το finite state machine. Το αποτέλεσμα περνάει από register για συγχρονισμό με το ρολόι και προς αποφυγήν glitches στην έξοδο.

Οι πόροι που χρησιμοποιήθηκαν για N=64 και N=128 είναι οι παρακάτω:



Resource	Utilization	Available	Utilization %
LUT	4272	17600	24.27
LUTRAM	455	6000	7.58
FF	6728	35200	19.11
BRAM	12.50	60	20.83
BUFG	2	32	6.25

Name	Slice LUTs (17600)	Block RAM Tile (60)	Bonded IOPADs (130)	BUFGCTRL (32)	Slice Registers (35200)	F7 Muxes (8800)	Slice (4400)	LUT as Logic (17600)	LUT as Memory (6000)
dvlsi2021_lab5_top	4272	12.5	130	2	6728	14	1925	3817	455
main_node (main_node_new_image_generator)	374	1.5	0	0		0	140	374	0
input_delay_reg1 (reg_8bit)	0	0	0	0		0	3	0	0
main_node_instance (main_node)	374	1.5	0	0		0	138	374	0
comp (computation_module)	13	0	0	0		0	18	13	0
fifo (fifo_with_registers)	297	1.5	0	0		0	120	297	0
fsm (finite_state_machine)	27	0	0	0		0	11	27	0
sync (synchronizer)	53	0	0	0		0	30	53	0
valid_out_reg (reg_1bit_0)	0	0	0	0		0	1	0	0
valid_delay_reg1 (reg_1bit)	0	0	0	0		0	1	0	0
PROCESSING_SYSTEM_INSTANCE (design_1_wrapper)	3898	11	0	2		14	1798	3443	455

Δεν υπάρχει κάποια διαφοροποίηση όταν αλλάζουμε το N, καθώς το μέγεθος των fifo που χρησιμοποιήθηκαν για την μετατροπή του σειριακού σε παράλληλο έχουν μέγεθος 2048 και καλύπτει όλες τις περιπτώσεις που θα μελετήσουμε χωρίς να προκαλείται υπερχείλιση. Ακόμη οι counters που χρησιμοποιήθηκαν για τον συγχρονισμό των wr_enable και rd_enable των 2 τελευταίων fifo παραμένουν ίδιοι και απλώς μεταβάλλεται η μέγιστη τιμή που μπορούν να πάρουν.

Για N=32 το latency είναι 68 κύκλοι. Πιο συγκεκριμένα, το valid_in χρειάζεται να περάσει από δύο register, προκειμένου να φτάσει στην είσοδο του φίλτρου, δηλαδή φτάνει μετά από 2 κύκλους. Στην συνέχεια χρειάζονται 32 κύκλοι μέχρι να αρχίσει η πρώτη fifo να δίνει δεδομένα στην έξοδο της και άλλοι 32 μέχρι να αρχίσει η δεύτερη fifo. Από την στιγμή που η δεύτερη fifo

βγάλει δεδομένο στην έξοδο της, χρειάζονται 2 κύκλοι προκειμένου αυτό το δεδομένο να φτάσει στην έξοδο του δεύτερου από τους τρεις registers που βρίσκονται στην έξοδο της fifo και το valid_out να γίνει 1.

Αντίστοιχα για N=64, όπου οι fifo δίνουν δεδομένα στην έξοδο όταν περιέχουν 64 δεδομένα, το latency είναι 2+64+64+2 = 132 κύκλοι.

Για N=128 το latency είναι 2+128+128+2= 260 κύκλοι.

Το throughput του συστήματος είναι 1, καθώς μετά το αρχικό latency θα δίνει μια έγκυρη έξοδο ανά κύκλο. Ακόμη, το σύστημα είναι συνεχούς διοχέτευσης, δηλαδή μπορεί να έρθει νέα εικόνα κατά τη διάρκεια της επεξεργασίας της προηγούμενης και θα γίνεται κανονικά η έισοδος των δεδομένων της στις fifo.