ΠΑΝΕΠΙΣΤΗΜΙΟ
ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

Ενσωματωμένα Συστήματα

Ενότητα: ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ Xilinx Pynq SBC (single board computer with FPGA)

Απαιτούμενος Αρ. Εργατοωρών: 16 (προσεγγιστικά)

Δρ. Μηνάς Δασυγένης

mdasyg@ieee.org

Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών

Εργαστήριο Ψηφιακών Συστημάτων και Αρχιτεκτονικής Υπολογιστών

http://arch.icte.uowm.gr/mdasyg

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Περιεχόμενα

1. Σκοπός της άσκησης

- Γνωριμία με την αναπτυξιακή πλακέτα Xilinx Pynq-Zq

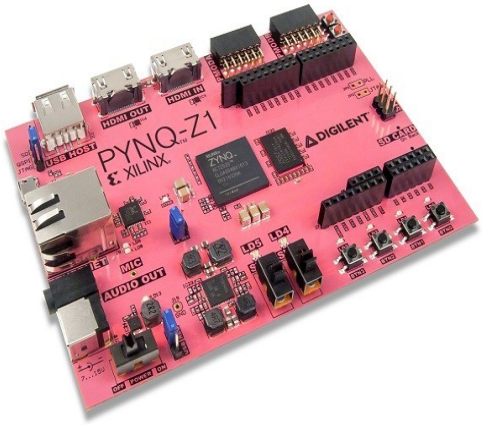
2. Περιγραφή της πλακέτας

Η πλακέτα Pynq (Εικόνα 1) είναι μια αναπτυξιακή πλακέτα στην οποία υπάρχει ένας διπύρηνος επεξεργαστής ARM που εκτελεί ένα λειτουργικό σύστημα Ubuntu Linux ειδικά διαμορφωμένο από τη Xilinx, ενώ ταυτόχρονα ο επεξεργαστής φέρει και ένα FPGA τμήμα, στο οποίο μπορεί κάποιος να υλοποιήσει Hardware Modules χρησιμοποιώντας VHDL ή Verilog ή άλλες γλώσσες που υποστηρίζει το αναπτυξιακό πρόγραμμα Xilinx Vidavo. Συγκεκριμένα η πλακέτα φέρει:

Feature	**Description**
CPU+FPGA	• **Zynq-7000 AP SoC XC7Z020-1CLG400C** • **Dual ARM® Cortex™-A9 MPCore™ with CoreSight™** • **32 KB Instruction, 32 KB Data per processor L1 Cache** • **512 KB unified L2 Cache** • **256 KB On-Chip Memory** • **2x UART, 2x CAN 2.0B, 2x I2C, 2x SPI, 4x 32b GPIO** • **2x USB 2.0 (OTG), 2x Tri-mode Gigabit Ethernet, 2x SD/SDIO on-chip peripherals** • **85K logic cells (13300 logic slices, each with four 6-input LUTs and 8 flip-flops)** • **630 KB of fast block RA** • **Four clock management tiles, each with phase-locked loop (PLL)** • **220 DSP slices** • **Internal clock speeds exceeding 450MHz** • **2x 12 bit, 1 MSPS On-chip analog-to-digital converter (XADC)**
I/O Interfaces	• USB-JTAG Programming circuitry • USB OTG 2.0 • USB-UART bridge • One 10/100/1G Ethernet • HDMI Input • HDMI Output • Electret microphone with pulse density modulated (PDM) output • 3.5mm mono audio output jack, pulse-width modulated (PWM) format
Memory	• 512 Mbyte DDR3

	• 128 Mbit Quad-SPI Flash • Micro SD card connector
Switches and LEDs	• 2 Slide switches • 2 RGB LEDs • 4 LEDs • 4 Push-buttons
Clocks	• One 125 MHz for PL • One 50 MHz for PS
Expansion ports	• 2 Pmod ports • 1 Arduino Shield • 16 GPIO • 6 Single-ended 0-3.3V Analog inputs to XADC • 4 Differential 0-1.0V Analog inputs to XADC

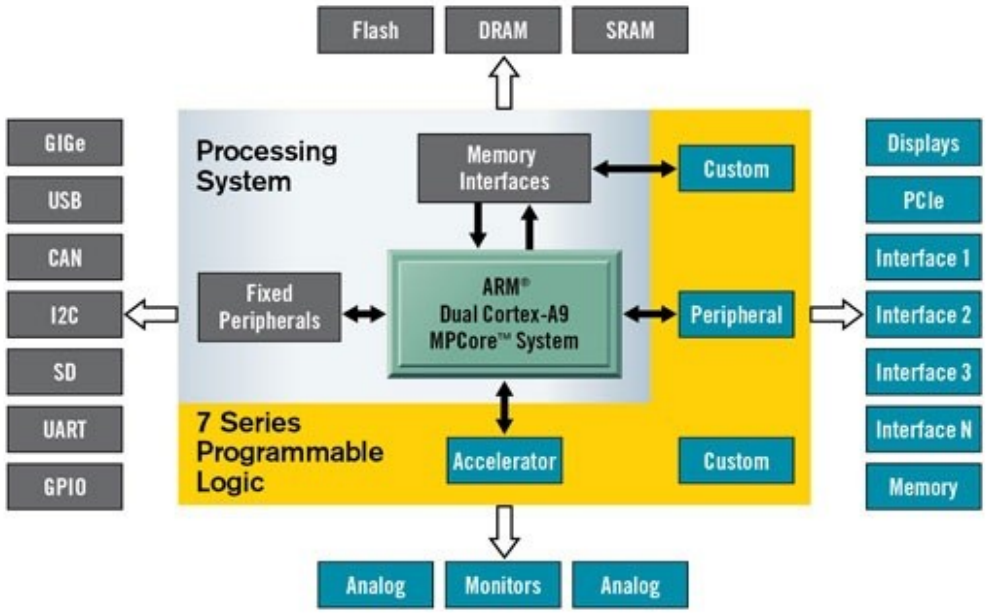
Το ιδιαίτερο χαρακτηριστικό της πλακέτας ως προς την παρόμοια πακέτα Xilinx Zedboard, είναι ότι ο προγραμματισμός μπορεί να γίνει μέσω Python, είναι πολύ πιο φιλική προς το χρήστη, αφού εκτελεί linux ubuntu και όχι petalinux, ενώ μπορεί να φορτώνει ρυθμίσεις προγραμματισμού FPGA (bitstream) κατά τη διάρκεια εκτέλεσης, χωρίς να απαιτείται επανεκκίνηση του συστήματος.



Εικόνα 1: Η αναπτυξιακή πλακέτα Xilinx Pynq-Z1

Όπως φαίνεται στην αρχιτεκτονική του Pynq (Εικόνα 2) κάποια περιφερειακά είναι άμεσα συνδεδεμένα στον επεξεργαστή ARM (fixed peripherals), ενώ κάποια άλλα συνδέονται στην προγραμματιζόμενη δομή FPGA (programmable logic ή PL). Αυτό σημαίνει ότι για να χρησιμοποιήσετε αυτά τα περιφερειακά, θα πρέπει να χρησιμοποιήσετε κάποιο ελεγκτή που έχει προγραμματιστεί σε HDL και έχει μεταφερθεί στο FPGA. Τα περισσότερα από αυτά τα περιφερειακά, έχουν ήδη κάποιες υλοποιήσεις στο λειτουργικό σύστημα, που αρκεί να τις φορτώσει κάποιος (όπως θα φανεί στις ασκήσεις που ακολουθούν). Η ανάπτυξη του αρχείου bistream προγραμματισμού του FPGA γίνεται στο αναπτυξιακό πρόγραμμα Xilinx Vidavo, συνήθως σε VHDL ή σε Verilog.

Εκτός από τα περιφερειακά, το FPGA χρησιμοποιείται για τη δημιουργία επιταχυντών επειδή το υλικό συνήθως εκτελείται πολύ πιο γρήγορα από ότι το λογισμικό, αφού μπορεί να επιτευχθεί ένας πολύ μεγάλος βαθμός παραλληλίας.



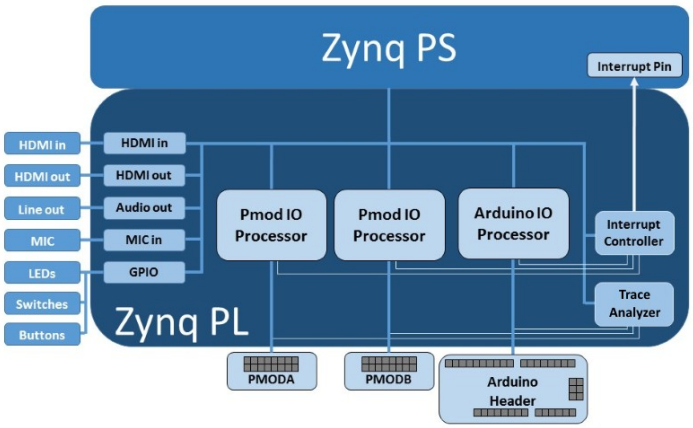
Εικόνα 2: Η αρχιτεκτονική του Xilinx Pynq

Η Εικόνα 3 παρουσιάζει τη σύνδεση της προγραμματιζόμενης λογικής με όλα τα περιφερειακά. Σε περίπτωση που απαιτείται η χρήση κάποιου περιφερειακού από αυτά, θα πρέπει να δημιουργηθεί και να φορτωθεί το κατάλληλο bitstream προγραμματισμού, το οποίο η Xilinx το ονομάζει επικάλυψη (overlay). Μάλιστα, μπορεί να χρησιμοποιηθεί και ένας προγραμματιζόμενος επεξεργαστής της Xilinx, όπως το Microblaze για να χειρίζεται αποκλειστικά τα περιφερειακά.

Τα παρακάτω περιφερειακά συνδέονται στο PL:

- HDMI είσοδος
- HDMI έξοδος με αναλύσεις: 640x480, 800x600, 1280x720 (720p), 1280x1024, 1920x1080 (1080p)*

- Ενσωματωμένο μικρόφωνο
- Έξοδος ήχου
- I/O (Είσοδος έξοδος) με: 2 τριών-χρωμάτων LED, 2 διακόπτες, 4 κουμπιά πίεσης, 4 ξεχωριστά LEDs.
- 2 αρχεία ρυθμίσεων για Microblaze: (a) Arduino Microblaze, και (b) PMOD Microblaze
- Αναλυτής ίχνους (trace analyzer), ο οποίος επιτρέπει την σύλληψη της επικοινωνίας από τις διεπαφές είσοδου εξόδου και την καταγραφή στη μνήμη DRAM, προκειμένου να μπορέσει κάποιος να εξετάσει την επικοινωνία με λεπτομέρεια, ίσως με συνδυασμό του πακέτου Wavedrom της Python.



Εικόνα 3: Η προγραμματιζόμενη δομή FPGA συνδέεται με συγκεκριμένα περιφερειακά

3. Σύνδεση με τη πλακέτα

Η σελίδα που μπορεί να προμηθευτεί κάποιος την πλακέτα, όπως και να κατεβάσει την τελευταία έκδοση του λειτουργικού συστήματος Linux ή το εργαλείο Vidavo για τη δημιουργία bitstreams (διαμορφώσεις FPGA για τον επαναπρογραμματισμό τους) είναι η https://www.xilinx.com/support/university/boards-portfolio/xup-boards/XUPPYNQ.htm

Η πλακέτα μπορεί να τροφοδοτηθεί μέσω του USB καλωδίου συνδεδεμένο στη micro USB θύρα PROG UART. Σε περίπτωση που υπάρχουν περιφερειακά στην πλακέτα που έχουν απαιτήσεις σε ρεύμα, τότε μπορεί να χρησιμοποιηθεί το εξωτερικό τροφοδοτικό 12V, 3 A που δίνεται μαζί με την πλακέτα.

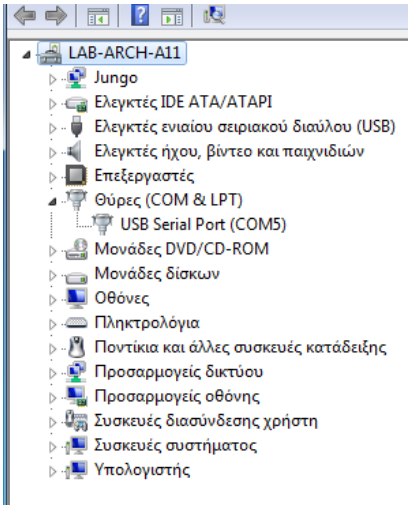
Επίσης, η πλακέτα απαιτεί το λειτουργικό σύστημα σε μια εξωτερική κάρτα SD. Η SD κάρτα που έρχεται μαζί με την πλακέτα φέρει το λειτουργικό σύστημα Ubuntu. Ο χρήστης μπορεί αν θέλει να κατεβάσει την τελευταία έκδοση από την παραπάνω

σελίδα στο δεσμό “Download PYNQ Image”. Η έκδοση που έχουν τα board αυτή τη στιγμή (2017) είναι η 2016/09/14 (αναγράφεται στο όνομα του αρχείου).

Αν πρόκειται να τροφοδοτήσετε την πλακέτα από το USB να θέσετε το jumper στη κάτω αριστερή γωνία στο ‘USB’. Αν θέλετε να τροφοδοτήστε την πλακέτα μέσω τροφοδοτικού DC, τότε θα πρέπει να θέσετε το jumper στην επιλογή ‘REG’.

Το board επιτρέπει σύνδεση console (ταυτόχρονα με την παροχή ρεύματος) με τη χρήση του καλωδίου microUSB – USB Type A.

=>Τοποθετήστε το καλώδιο στο board και στον Windows υπολογιστή σας (που έχει εγκατεστημένο το Xilinx Vidavo). Περιμένετε να ολοκληρωθεί η εγκατάσταση των οδηγών. Στη συνέχεια, από το Device Manager, δείτε τη COM θύρα που έχει δημιουργηθεί (Εικόνα 4).



Εικόνα 4: Μόλις ολοκληρωθεί η εγκατάσταση των οδηγών θα εμφανιστεί μια νέα USB Serial Port στο Device manager

=>Ανοίξτε το putty, επιλέξτε σύνδεση Serial, ταχύτητα 115200 και πατήστε Open.

Υπάρχει ο χρήστης xilinx με κωδικό xilinx. Ο χρήστης μπορεί να γίνει root για την εγκατάσταση πακέτων με την εντολή `sudo bash` . Στην προτροπή για κωδικό δώστε τον κωδικό xilinx .

=>Τοποθετήστε ένα καλώδιο LAN στο board (αν δεν το έχετε ήδη συνδέσει) που συνδέεται στο τοπικό δίκτυο σας. Επιβεβαιώστε ότι έχει πάρει IP με την εντολή `ifconfig`

=> Επιβεβαιώστε τη συνδεσιμότητα του board σας στο διαδίκτυο. Δοκιμάστε αν υπάρχει IP connectivity με την εντολή:

```
ping 83.212.19.218
```

πατήστε CONTROL+C μετά από λίγα δευτερόλεπτα και δείτε αν έχετε λάβει απάντηση. Αν δεν έχετε λάβει απάντηση να το διερευνήσετε με το διδάσκοντα.

Δοκιμάστε αν υπάρχει DNS IP connectivity με την εντολή:

```
ping zafora.icte.uowm.gr
```

πατήστε CONTROL+C μετά από λίγα δευτερόλεπτα και δείτε αν έχετε λάβει απάντηση. Αν δεν έχετε λάβει απάντηση να το διερευνήσετε με το διδάσκοντα.

Αν έχετε λάβει απάντηση και στις 2 περιπτώσεις, μπορείτε να συνεχίσετε στο επόμενο βήμα.

=> Εκτελέστε **`apt-get update ; apt-get upgrade`** για την αναβάθμιση των πακέτων της πλακέτας. Μάλιστα, αν θέλετε να προχωρήσετε σε major upgrade μπορείτε να εκτελέσετε και την εντολή **`apt-get dist-upgrade`**.

Επειδή υπάρχει η περίπτωση να μη μπορεί να συνδεθεί στα repositories που φέρει το image, μπορείτε να χρησιμοποιήσετε τις παρακάτω οδηγίες για να αντικαταστήσετε τα αποθετήρια με τα παλιά για να κάνετε τις εγκαταστάσεις των πακέτων:

1) modify with sudo the file /etc/apt/sources.list

2) comment all lines

3) insert the following lines:

```
deb http://old-releases.ubuntu.com/ubuntu/ wily main universe restricted multiverse

deb-src http://old-releases.ubuntu.com/ubuntu/ wily main universe restricted multiverse


deb http://old-releases.ubuntu.com/ubuntu/ wily-security main universe restricted multiverse

deb-src http://old-releases.ubuntu.com/ubuntu/ wily-security main universe restricted multiverse


deb http://old-releases.ubuntu.com/ubuntu/ wily-updates main universe restricted multiverse

deb-src http://old-releases.ubuntu.com/ubuntu/ wily-updates main universe restricted multiverse
```

3) save the file

4) do: sudo apt-get update

4. Το πρώτο αναπτυξιακό πρόγραμμα της πλακέτας

Η πλήρης τεχνική τεκμηρίωση της πλακέτας βρίσκεται στον ιστοχώρο:

http://pynq.readthedocs.io/en/v2.0/

και στο

https://reference.digilentinc.com/reference/programmable-logic/pynq-z1/start

Επίσης, υπάρχουν εισαγωγικές οδηγίες (για άτομα χωρίς μεγάλη εμπειρία στον προγραμματισμό) στον ιστοχώρο:

http://www.pynq.io/

Το PYNQ χρησιμοποιεί ως βασικό εργαλείο χρήσης το Jupiter Notebook, που είναι ένας διαδραστικός ιστοχώρος που εκτελείται πάνω στο board και παρέχει εύκολο προγραμματισμό σε python, μέσω ενός οποιουδήποτε σύγχρονου browser.

Μπορείτε να αλλάξετε το όνομα της συσκευής, είτε με τις γνωστές εντολές του ubuntu (επεξεργασία των αρχείων `/etc/hostname` , `/etc/hosts`) είτε με το να συνδεθείτε ως xilinx χρήστης και να δώστε:

```
cd ~/scripts
```

```
sudo ./hostname.sh το_νέο_όνομα
```

Για να μεταφέρετε αρχεία από και προς την πλακέτα μπορείτε να χρησιμοποιήσετε (όνομα χρήστη: xilinx, κωδικός xilinx) το SCP ή το CIFS με το να δώσετε **`\\pynq`** ή **`\\IP_of_PYNQ`**

4.1 Τρόποι Ανάπτυξης Υλικού και Λογισμικού στο board

- (1) Μπορεί να χρησιμοποιηθεί python τόσο μέσω του Jupiter Notebook όσο και μέσω τερματικού.
- (2) Μπορεί να χρησιμοποιηθεί κάποια γλώσσα περιγραφής υλικού, όπως η VHDL στο Vidavo Design Tools.
- (3) Μπορεί να χρησιμοποιηθεί C/C++ μέσω του SDK που παρέχεται από το Vidavo DesignTools.

4.1.1 Χρήση του Juputer Notebook

Το πιο απλό είναι να χρησιμοποιηθεί η Python με τις βιβλιοθήκες που παρέχουν έτοιμα FPGA configurations για τα διάφορα modules του συστήματος (π.χ. για την HDMI είσοδο ή HDMI έξοδο).

Συνδεθείτε στην πλακέτα που έχει συνδεθεί στο δίκτυο. Δώστε την εντολή εύρεσης της IP που έχει λάβει (π.χ.):

```
ifconfig eth0
```

Στον host υπολογιστή ανοίξτε ένα φυλλομετρητή και μεταβείτε στην IP στη θύρα 9090. Π.χ. αν η IP είναι 192.168.70.127 ανοίξτε τη σελίδα

http://192.168.70.127:9090

Αν σας ζητηθεί κωδικός, τοποθετήστε **`xilinx`** .

Εναλλακτικά, μπορείτε να εκτελέσετε ως root στο τερματικό το **`python3`** και να αναπτύξετε τον κώδικά σας εκτός του φυλλομετρητή. Να προτιμήσετε όμως την πρώτη φορά το φυλλομετρητή, γιατί είναι πολύ πιο φιλικό στη χρήση.

Από τη σελίδα που εμφανίζεται, μπορείτε να δείτε δοκιμαστικούς κώδικες και να τους εκτελέσετε άμεσα από τη σελίδα σας (οι κώδικες εκτελούνται στην πλακέτα, και η έξοδος εμφανίζεται στο φυλλομετρητή).

Εισέλθετε στον κατάλογο Getting_Started .

1. Διαβάστε το 1_jupyter_notebook για να καταλάβετε τη χρήση του Jupyter_notebook (αν δεν το έχετε συναντήσει ή χρησιμοποιήσει ποτέ ξανά).

2. Στη συνέχεια εκτελέστε το 2_programming_python και στο notebook που θα σας εμφανιστεί, δείτε και μελετήστε τους κώδικες python και στη συνέχεια εκτελέστε τους (θα δείτε τα αποτελέσματα στην ίδια σελίδα). Τροποποιήστε τμήματα του κώδικα και ξανα-εκτελέστε τα για να καταλάβετε καλύτερα τη λειτουργία.

3. Εκτελέστε το 3_programming_onboard για να δείτε πως μπορείτε να χειριστείτε τα περιφερειακά (switches/leds/RGB/buttons). Σημειώστε ότι μπορείτε να εκτελέστε τον κώδικα python που αναγράφεται κατευθείαν στο τερματικό. Με το notebook όμως δε χρειάζεται να συνδέεστε στο τερματικό και μπορείτε να ανακαλύπτετε τις δυνατότητες μέσω του browser (δηλαδή, είναι πιο φιλικό στο χρήστη). Ασφαλώς, ένα τελικό προϊόν δε θα χρησιμοποιεί το browser, αλλά θα εκτελεί άμεσα τα scripts από το τερματικό.

   Κατά διαστήματα να πηγαίνετε στην αρχική σελίδα του notebook και να σκοτώνετε (shutdown) τα notebook που δε χρειάζονται για να ελευθερώνετε τους πόρους του συστήματος.

4. Δείτε το 4_base_overlay_iop για τη χρήση των επιπρόσθετων διασυνδέσεων. Όμως, δεν υπάρχουν στο εργαστήριο περιφερειακά για να συνδέσουμε, και έτσι απλώς μόνο διαβάστε το.

5. Ανοίξτε το 5_base_overlay_video και δείτε πως μπορείτε να κάνετε capture HDMI και πως μπορείτε να στείλετε έξοδο σε HDMI. Μπορείτε να χρησιμοποιήστε και έξτρα HDMI οθόνη για να διαπιστώσετε τη λειτουργία του.

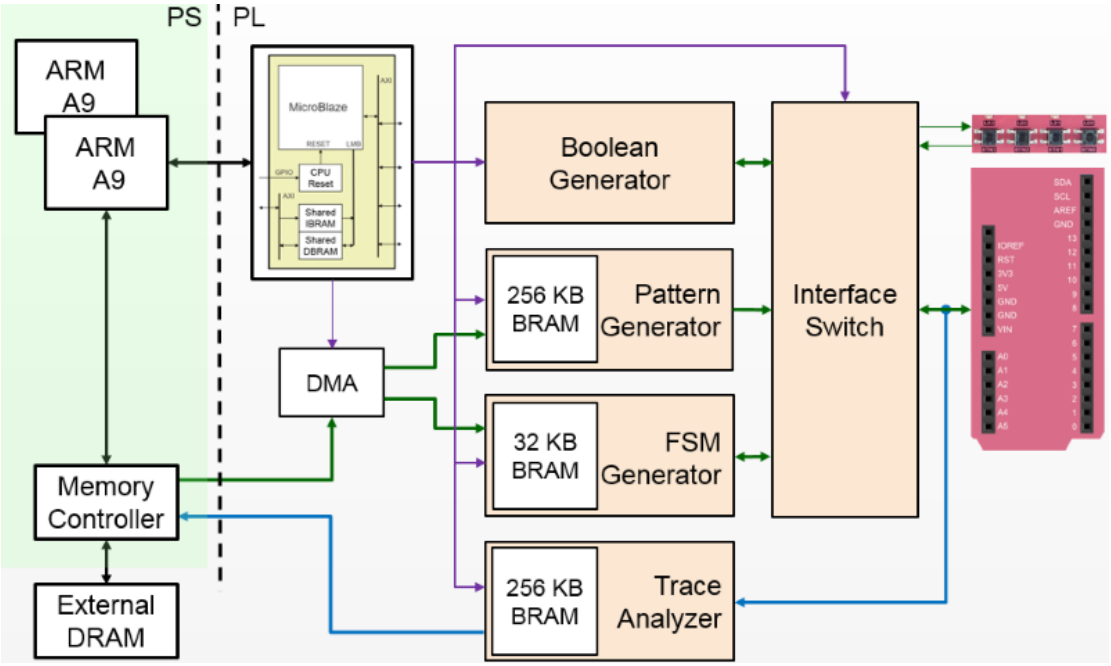
6. Ανοίξτε το 6_base_overlay_audio και δείτε πως μπορείτε να χειριστείτε τον ήχο. Συγκεκριμένα, πως να κάνετε capture τον ήχο μέσω του ενσωματωμένου μικροφώνου και πως μπορείτε να τον αναπαράγετε στην αντίστοιχη διεπαφή.

Μετά την εισαγωγή στο board θα μεταβείτε στα Examples όπου θα δείτε πιο σύνθετες υλοποιήσεις που χρησιμοποιούν το board, όπως χρήση του OpenCV για όραση μηχανής. Μεταβείτε στον φάκελο Examples, αφού τερματίσετε όλα τα υπόλοιπα kernels.

Τα πιο σημαντικά examples που πρέπει να μελετήσετε είναι:

- arduino_analog
- audio_playback
- board_btns_leds
- opencv_face_detect_hdmi
- opencv_face_detect_webcam
- opencv_filters_hdmi
- opencv_filters_webcam
- overlay_download
- overlay_integration
- python_random
- python_retrieving_shell
- usb_webcam
- usb_wifi
- video_filters
- tracebuffer_i2c

Μια χρήσιμη επικάλυψη υλικού, είναι επίσης το logictools (Εικόνα 5) (http://pynq.readthedocs.io/en/latest/pynq_overlays/logictools_overlay.html), το οποίο επιτρέπει τη χρήση FSM, Γεννήτρια Boolean τιμών, τη Γεννήτρια προτύπων, ανάλυση ίχνους και άλλα μπλοκ υλικού να συνδεθούν στους ακροδέκτες τύπου arduino. Η όλη διαχείριση γίνεται μέσω ενός soft-core επεξεργαστή Microblaze που έχει περιγραφεί και ενσωματώνεται στην επικάλυψη.



Εικόνα 5: Η επικάλυψη logictools ρυθμίζει κατάλληλα το υλικό για τη σύνδεση και αποσφαλμάτωση εξωτερικών συσκευών εισόδου/εξόδου

4.1.2 Χρήση του Αναπτυξιακού περιβάλλοντος Vivado Design Tools με HDL

Σημειώσεις:

Για να μεταφέρετε δεδομένα ανάμεσα στον επεξεργαστή ARM και την προγραμματιζόμενη δομή FPGA να ανατρέξετε στην παράγραφο **Numpy Data Movement** στο URL: http://pynq.readthedocs.io/en/latest/python_environment.html .

Περιληπτικά, θα πρέπει να καλέσετε τη συνάρτηση `pynq.Xlnk().cma_array(shape, dtype)` , ενώ μπορείτε να δείτε και τη διεύθυνση που έχει γίνει η αντιστοίχηση (ώστε να τη στείλετε στη δομή FPGA) με την εντολή `hex(buffer.physical_address)` .

Επίσης, μπορείτε να χρησιμοποιήσετε την γεγονοδηγούμενη τεχνική επικοινωνίας μ διακοπές (interrupts), ώστε ο επεξεργαστής να μπορεί να απελευθερωθεί για να εκτελεί κάποιο άλλο έργο, και όταν ολοκληρωθεί η επεξεργασία να στέλνει η προγραμματιζόμενη FPGA δομή ένα event (δηλαδή, μια διακοπή). Για να δείτε πως μπορείτε να υλοποιήσετε μια τέτοια επικοινωνία, μελετήστε την παράγραφο **Asyncio Integration** στο URL: http://pynq.readthedocs.io/en/latest/python_environment.html .

Περιληπτικά, θα πρέπει να χρησιμοποιήσετε τη βιβλιοθήκη **asyncio** , να θέσετε μια εντολή **await** για να περιμένει την ολοκλήρωση ενός event και να δημιουργήσετε την κατάλληλη εργασία με το **create_task()**.

Άσκηση για το σπίτι

Μπορείτε **να κάνετε και άλλες ασκήσεις σε αυτή την πλακέτα που βρίσκονται στη διεύθυνση: https://github.com/Xilinx/PYNQ_Workshop**

Άσκηση για το σπίτι

Δημιουργήστε ένα sample bitstream configuration για την πλακέτα, χρησιμοποιώντας το Vidavo.

Άσκηση για το σπίτι

Προσπαθήστε να αναβοσβήσετε ένα LED που έχετε συνδέσει κατάλληλα στο board. Μπορείτε να ξεκινήσετε από τον παρακάτω κώδικα και να προβείτε στις κατάλληλες διορθώσεις:

...

```
from pynq.iop import ARDUINO
from pynq.iop import Arduino_IO
from pynq import Overlay
ol = Overlay("base.bit")
ol.download()
pin0=Arduino_IO(ARDUINO,0,"out")
```

4.1.3 Χρήση του Αναπτυξιακού περιβάλλοντος Vivado Design Tools με SDK C/C++

Άσκηση για το σπίτι

Δημιουργήστε μια απλή εφαρμογή C/C++ στο SDK του Vidavo Design. Μπορείτε να δείτε τα παραδείγματα:

- https://github.com/beja65536/pz1_sobelfilter
- https://github.com/hackwa/pynqfire