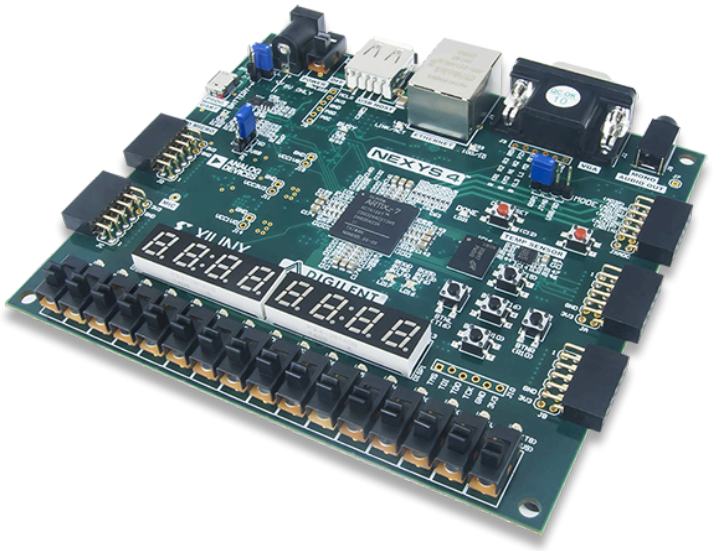
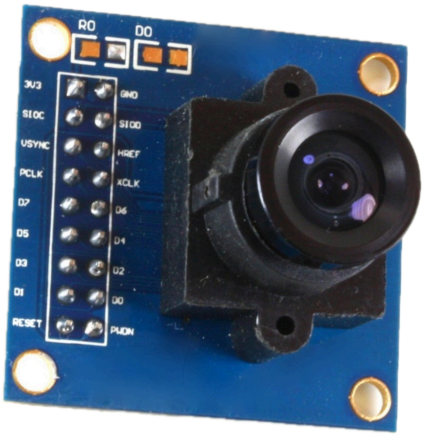
FPGA-based

Video Transmission &

Image Processing Project





1. Echipa noastră:

- Ciobanu Daria
- Doca Andrei
- Georgescu Cătălin
- Hondola Paul
- Prof. Dr. Ing. Amaricai Boncalo Oana

2. Task-ul nostru:

- Transmisia video în timp real a datelor de pe o cameră și afișarea pe un display prin intermediul unui port VGA
- Recepționarea datelor transmise prin intermediul protocolului UART și afișarea pe un display prin intermediul unui port VGA

3. Specificații sistem

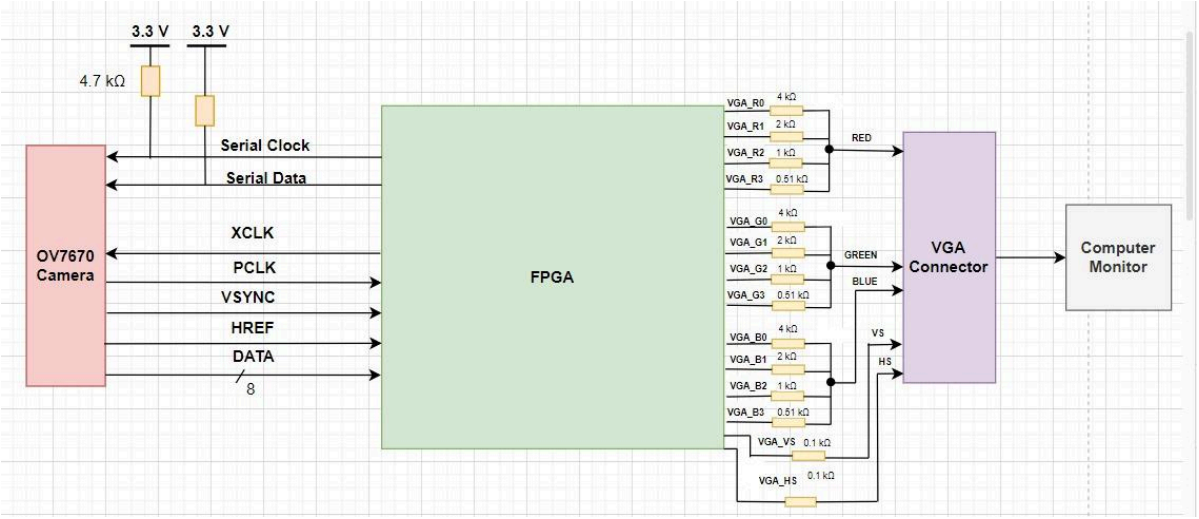
- FPGA: Xilinx Nexys 4 Artix-7 (Digilent)
- Camera: OV7670 pentru video în timp real
- Monitor cu port VGA
- IDE: AMD Vivado 2023.2
- Limbaj: Verilog + Python

4. Arhitectura sistemului

4.1. FPGA

- 100 MHz Clock
- PMOD JA / JB
- Port VGA
- UART

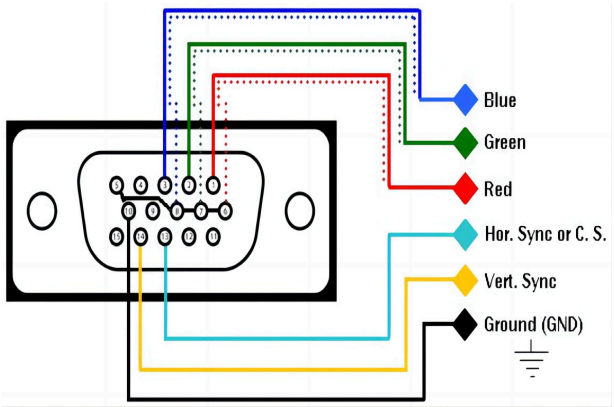
4.2. TOP LEVEL



4.3. VGA

Primul pas al proiectului nostru a fost configurarea controller-ului VGA (Video Graphics Array).

Connector-ul VGA

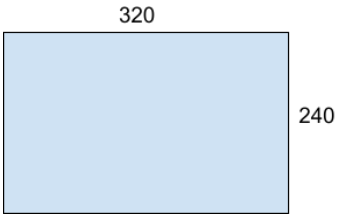


În imaginea de mai sus se observă pinii conectorului VGA

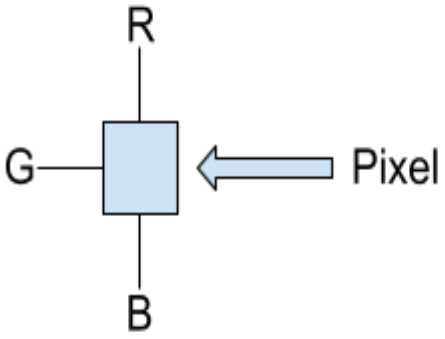
- BLUE, GREEN, RED - Culorile transmise de conector, in general pe **4** biti [3:0];
- Horizontal Sync (Hsync) - Semnalul de sincronizare pe orizontală, acesta semnalează cand controller-ul trebuie să meargă pe următoarea linie pe orizontală.

- Vertical Sync (Vsync) - Semnalul de sincrionizare pe verticala, acesta semnalează cand se da refresh la imagine.

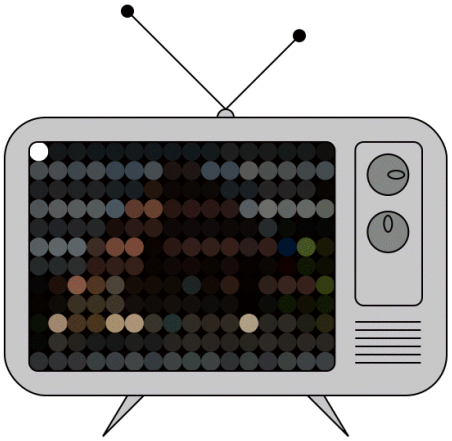
Sa zicem că avem o imagine de 320



Asta înseamnă ca avem 320*240(76800) de pixeli, fiecare pixel are o culoare RGB (pentru simplitate folosim RGB444).

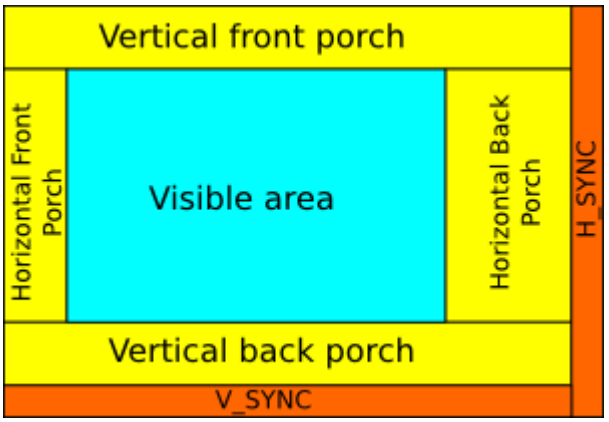


În procesul de afișare a unei imagini, fiecare pixel trebuie să primească o culoare specifică pentru a compune imaginea finală. Pentru a realiza acest lucru, imaginea trebuie parcursă linie cu linie și pixel cu pixel. Sincronizarea și stabilitatea afișajului sunt esențiale pentru a evita distorsiunile și artefactele vizuale.



Aici intervin elementele de bază ale sincronizării semnalului video: front porch, back porch și pulse. Acestea asigură că fiecare linie și fiecare cadru sunt afișate corect și la momentul potrivit.

- **Front Porch** - reprezintă intervalul de timp dintre sfârșitul unei linii active de imagine și începutul impulsului de sincronizare orizontală.
- **Back Porch** - este intervalul de timp dintre sfârșitul impulsului de sincronizare orizontală și începutul liniei active de imagine.
- **Pulse** - este un semnal scurt trimis pentru a sincroniza începutul unei noi linii sau cadre. Impulsul de sincronizare orizontală marchează începutul unei noi linii, în timp ce impulsul de sincronizare verticală marchează începutul unui nou cadru.



Sa trecem la implementarea pe FPGA in verilog, pentru acest lucru trebuie sa luăm în considerare cateva aspecte:

- **Ce rezoluție vrem să afișăm?** - Rezoluția determină numărul total de pixeli care trebuie generați și sincronizați pentru fiecare cadru de imagine
- **Ce frecvență la clock trebuie să folosim?** - Frecvența ceasului FPGA trebuie să fie suficient de mare pentru a permite generarea și sincronizarea corectă a tuturor pixelilor la rezoluția și rata de refresh dorite
- **Ce refresh rate vrem sa avem?**

Rezolutia, frecvența ceasului sunt strâns legate

$$\text{Frecvența Ceasului} = \frac{\text{Rezoluție Orizontală Totală} \times \text{Rezoluție Verticală Totală} \times \text{Rată de Reîmprospătare}}{1 \text{ secundă}}$$

Rezolutie totala = rezolutie + front porch + back porch + pulse;

Exemplu:

- **Rezolutie:** 640x480
- **Refresh rate:** 30Hz

Pentru fiecare rezolutie, sunt standardizate valorile pentru Front Porch, Back Porch si Pulse.

Pentru rezolutia noastra:

- **Front Porch Orizontal:** 16
- **Back Porch Orizontal:** 48
- **Puls orizontal:** 96

- **Front Porch Vertical:** 10
- **Back Porch Vertical:** 33
- **Puls vertical:** 2

Rezolutie Totala Orizontala = 640 + 16 + 48 + 96 = 800

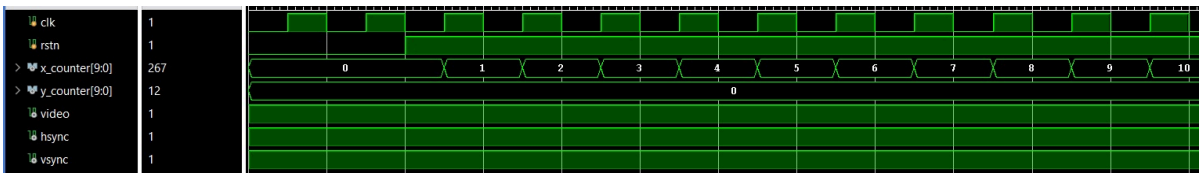
Rezolutie Totala Verticala = 480 + 10 + 33 + 2 = 525

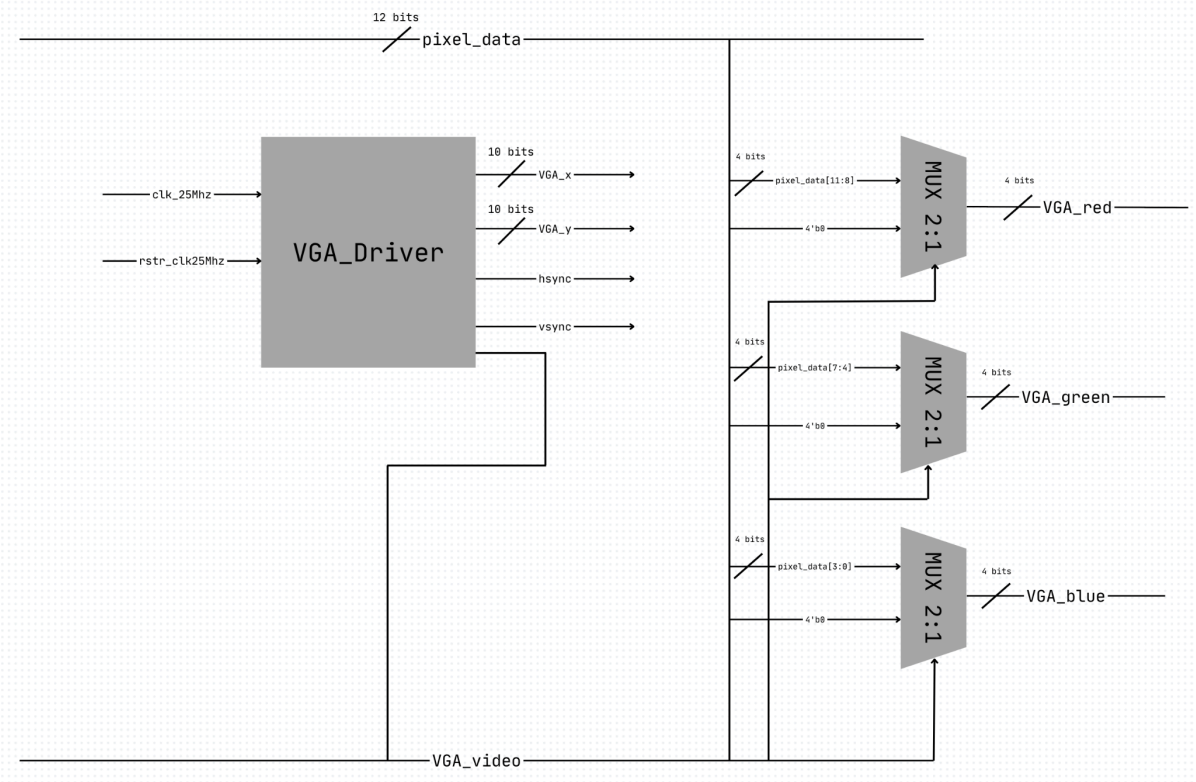
Frecventa Minima Ceas = 800*600*30 = 14 MHZ

Cod Verilog:

```
always @(posedge clk or negedge rstn)
begin
    if (!rstn)
    begin
        hc <= 0;
        vc <= 0;
    end
    else
    begin
        if (hc == hEND - 1)
        begin
            hc <= 0;
            if (vc == vEND - 1)
                vc <= 0;
            else
                vc <= vc + 1;
        end
        else
            hc <= hc + 1;
    end
```

vc = vertical_counter, hc = horizontal_counter;





Pentru implementarea noastră:

Show BlueScreen on VGA:
https://github.com/LD-FPGA-Project/FPGA-Image-Processing/tree/main/Src/VGA/Show-Blue_Screen

Show image on VGA:

https://github.com/LD-FPGA-Project/FPGA-Image-Processing/tree/main/Src/VGA/Show-Image

4.4. Camera OV7670

Dupa implementarea VGA-ului, următorul lucru este implementarea camerei.

OV7670 este un modul de cameră destul de complicat, dar cu o configurare corectă și o înțelegere detaliată a funcționării sale, poate fi utilizat pentru a captura imagini.

4.4.1. Configurarea Camerei:

- **OV7670** necesita configurarea printr-o interfață de comunicare similara cu I2C, numita SCCB(Serial Camera Control Bus). Aceste registre controlează diverse aspecte ale funcționării camerei, cum ar fi rezoluția, formatele de culoare, etc…

Address (Hex)	Register Name	Default (Hex)	R/W	Description
8C	RGB444	00	RW	Bit[7:2]: Reserved Bit[1]: RGB444 enable, effective only when COM15[4] is high 0: Disable 1: Enable Bit[0]: RGB444 word format 0: xR GB 1: RG Bx

4.4.2 Interfața cu FPGA

OV7670 are mai multe output-uri care trebuie captate de FPGA:

- **PCLK (Pixel Clock):** Este semnalul de ceas pentru transferul de date ale pixelilor. Fiecare ciclu de ceas PCLK corespunde unui nou pixel.
- **VSYNC (Vertical Sync):** Semnalul de sincronizare verticală indică începutul unui nou cadru.
- **HSYNC (Horizontal Sync)**: Semnalul de sincronizare orizontală indică începutul unei noi linii de scanare.

- **DATA[7:0]:** Linii de date care conțin informațiile de culoare ale pixelilor. În funcție de modul de operare, acestea pot transmite diferite formate de culoare (de exemplu, RGB565, YUV).
- **XCLK (External Clock):** Este semnalul de ceas extern furnizat de FPGA către camera OV7670 pentru funcționare.

Pentru documentatie completa:
https://www.fpga4student.com/2018/08/basys-3-fpga-ov7670-camera.html

Implementarea noastra :

PIN-OUT pentru PMOD

Acestea sunt denumirile conexiunilor de la cameră la porturile PMOD de pe placă.

Denumirile de la cameră se află pe partea frontală, sub obiectiv.

Conexiunile de pe placă sunt în diagrama de mai jos, pin-ul 1 are un semn pe placă lângă fiecare PMOD.

JA sau JB sunt grupările de pin-uri iar [x] este pinul de pe acel PMOD, urmând regula din diagramă.

NR.	**CAMERA**	**PMOD**

1	PWDN	JA[1]
2	D0	JA[2]
3	D2	JA[3]
4	D4	JA[4]
5	RESET	JA[7]
6	D1	JA[8]
7	D3	JA[9]
8	D5	JA[10]
9	D6	JB[1]
10	XCLK	JB[2]
11	HS (hsync)	JB[3]
12	SIOD (SDA)	JB[4]
13	D7	JB[7]
14	VS (vsync)	JB[8]

15	SIOC (SCL)	JB[9]
16	PCLK	JB[10]

4.5. UART (Universal Asynchronous Receiver Transmitter)

Pentru o functionalitate mai mare, am implementat protocolul UART.

Acest modul este un controler pentru transmiterea serială UART. Modulul gestionează stările necesare pentru a trimite date în format UART, controlând semnalul de transmitere (UART_TX) și semnalul de pregătire (READY).

Interfața modulului

Intrări

- **SEND** (wire): Semnal de comandă pentru a începe transmiterea.
- **DATA** (wire [7:0]): Datele de transmis (8 biți).
- **CLK** (wire): Ceasul sistemului.

Ieșiri

- **READY** (reg): Indică faptul că modulul este pregătit să primească date noi pentru transmitere.
- **UART_TX** (reg): Semnalul de ieșire serial UART.

Parametri

- **RDY**: Starea de așteptare (00).
- **LOAD_BIT**: Starea de încărcare a bitului următor (01).
- **SEND_BIT**: Starea de trimitere a bitului curent (10).

- **BIT_TMR_MAX**: Valoarea maximă a contorului de timp pentru fiecare bit (10415).
- **BIT_INDEX_MAX**: Numărul total de biți de transmis (10).

Registre interne

- **bitTmr** (reg [13:0]): Contorul pentru numărarea timpului pentru fiecare bit.
- **bitDone** (wire): Semnal care indică terminarea contorului de timp pentru un bit.
- **bitIndex** (reg [3:0]): Indexul curent al bitului transmis.
- **txBit** (reg): Valoarea bitului curent de transmis.
- **txData** (reg [9:0]): Datele de transmis, inclusiv start bit și stop bit.
- **txState** (reg [1:0]): Starea curentă a controlerului.

Funcționalitate

Stări ale controlerului

- **RDY**: Modulul așteaptă un semnal de comandă (SEND). Când SEND devine activ, controlerul trece la starea LOAD_BIT.
- **LOAD_BIT**: În această stare, datele sunt încărcate și controlerul trece la starea SEND_BIT.
- **SEND_BIT**: Bitul curent este trimis. Dacă bitul curent a fost transmis complet (bitDone este activ), controlerul verifică dacă toți biții au fost trimiși (bitIndex == BIT_INDEX_MAX). Dacă da, revine la starea RDY, altfel se întoarce la LOAD_BIT.

Gestionarea contorului de timp pentru biți

- **bitTmr**: Este resetat în starea RDY și la finalul fiecărui bit (bitDone).
- **bitDone**: Devine activ atunci când bitTmr ajunge la valoarea BIT_TMR_MAX.

Gestionarea indexului de biți

- **bitIndex**: Este resetat în starea RDY și incrementat în starea LOAD_BIT.

Încărcarea datelor

- **txData**: Este încărcat cu datele de transmis (DATA), împreună cu start bit (0) și stop bit (1), atunci când SEND devine activ.

Generarea semnalului UART_TX

- **txBit**: Este setat la 1 în starea RDY și la valoarea bitului curent (txData[bitIndex]) în starea LOAD_BIT.
- **UART_TX**: Este actualizat continuu cu valoarea txBit.

Generarea semnalului READY

- **READY**: Este activ (1) atunci când controlerul este în starea RDY.

Modulul tx_top este un controler de transmisie UART care primește o valoare de 32 de biți (i_display_value), o convertește în format ASCII și o trimite prin linia de transmisie UART (UART_TXD). Modulul utilizează un submodul UART_TX_CTRL pentru a gestiona procesul de transmisie UART.

Interfața modulului

Intrări

- **i_top_clk** (wire): Semnalul de ceas al sistemului.
- **i_top_rst** (wire): Semnalul de resetare.
- **i_display_value** (wire [31:0]): Valoarea de afișat, care va fi convertită și transmisă.

Ieșiri

- **UART_TXD** (wire): Linia de transmisie UART.

Definiții și constante

- **RST_REG**: Starea de resetare (0).
- **LD_INIT_STR**: Starea de încărcare a șirului inițial (1).
- **SEND_CHAR**: Starea de trimitere a unui caracter (2).
- **RDY_LOW**: Starea de așteptare pentru semnalul de pregătire să fie scăzut (3).
- **WAIT_RDY**: Starea de așteptare pentru semnalul de pregătire să fie ridicat (4).
- **RESET_CNTR_MAX**: 200,000 cicluri de ceas, folosit pentru contorul de resetare.

Registre interne și fire

- **uartState** (reg [2:0]): Starea curentă a mașinii de stare.
- **sendStr** (reg [7:0] [0:31]): Buffer pentru șirurile de caractere, extins pentru numere mai lungi.
- **strIndex** (reg [7:0]): Indexul curent în sendStr.
- **strEnd** (reg [7:0]): Indexul de sfârșit al șirului în sendStr.
- **reset_cntr** (reg [17:0]): Contorul de resetare.
- **uartRdy** (wire): Semnalul de pregătire de la UART_TX_CTRL.
- **uartSend** (reg): Semnalul de comandă pentru trimitere.
- **uartData** (reg [7:0]): Datele care sunt trimise.
- **uartTX** (wire): Linia de transmisie UART.

Logica contorului de resetare

Resetarea contorului se face la fiecare ciclu de ceas, cu excepția cazului în care contorul a atins valoarea RESET_CNTR_MAX sau starea nu este RST_REG.

Logica mașinii de stare UART

Stările principale

- **RST_REG**: Așteaptă până când contorul de resetare atinge valoarea maximă, apoi trece la starea LD_INIT_STR.
- **LD_INIT_STR**: Convertește valoarea de afișat în ASCII și o încarcă în sendStr, apoi trece la SEND_CHAR.
- **SEND_CHAR**: Trimite caracterul curent, apoi trece la RDY_LOW.
- **RDY_LOW**: Așteaptă până când semnalul de pregătire este scăzut, apoi trece la WAIT_RDY.
- **WAIT_RDY**: Așteaptă până când semnalul de pregătire este ridicat. Dacă toți caracterele au fost trimise, revine la LD_INIT_STR, altfel trimite următorul caracter.

Logica de numărare a caracterelor și trimitere

Actualizează indexul de caractere strIndex și semnalul uartSend pentru a controla trimiterea datelor.

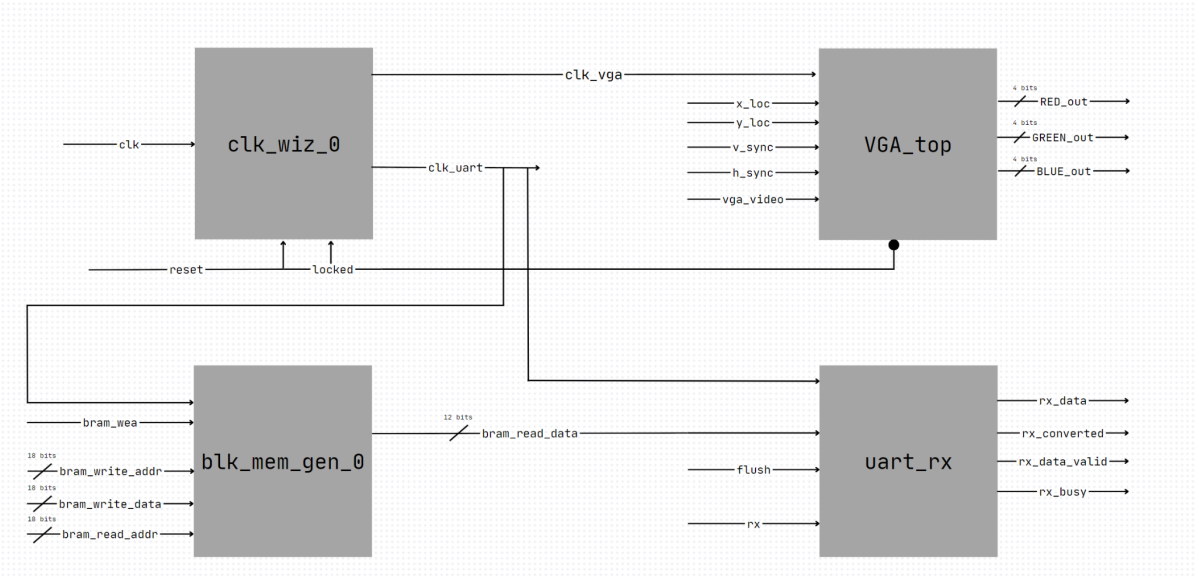
Instanțierea modulului UART_TX_CTRL

Modulul UART_TX_CTRL este instanțiat pentru a controla trimiterea caracterelor prin linia UART.

```
UART_TX_CTRL Inst_UART_TX_CTRL (
        .SEND(uartSend),
        .DATA(uartData),
        .CLK(i_top_clk),
        .READY(uartRdy),
        .UART_TX(uartTX)
);
```

assign UART_TXD = uartTX;

Block diagram



4.6. BRAM

…. de la camera OV7670 și furnizarea acestor date către modulul VGA pentru afișare. Să detaliem funcționarea acestui modul în contextul întregului sistem.

- **Scrierea Datelor**: Datele pixelilor capturate de camera OV7670 sunt stocate în memorie sincronizat cu clock-ul de pixeli (pclk). Adresa de memorie (o_pix_addr) specifică locația unde fiecare pixel este stocat.
- **Citirea Datelor**: Pentru afișare, datele sunt citite din memorie sincronizat cu clock-ul de 25 MHz (clk25m). Adresa de citire (o_VGA_pix_addr) este generată de modulul VGA pentru a accesa pixelii corecți din memorie.

Acest mecanism permite decuplarea temporară a procesului de capturare a imaginii de la camera și procesul de afișare pe display-ul VGA, permițând astfel un flux de date continuu și eficient între captură și afișare.

Modulul **mem_bram** din block diagram joacă un rol crucial în stocarea temporară a datelor de pixeli capturate de la camera OV7670 și furnizarea acestor date către modulul VGA pentru afișare. Să detaliem funcționarea acestui modul în contextul întregului sistem.

Funcționarea mem_bram

- **Recepția Datelor de la cam_capture**
  - **Intrări**:
    - **pclk**: Acesta este clock-ul de pixeli generat de camera OV7670. Fiecare ciclu de clock indică un nou pixel capturat de camera.
    - **o_pix_data[11:0]**: Aceasta este magistrala de date pe 12 biți care conține datele pixelului capturat (valori de culoare).
    - **o_pix_addr[18:0]**: Aceasta este adresa pe 19 biți care indică locația specifică în memorie unde trebuie stocat pixelul curent.
  - **Funcționalitate**:
    - În timpul fiecărui ciclu de clock **pclk**, datele de la camera (o_pix_data) sunt scrise în locația de memorie specificată de **o_pix_addr**. Astfel, mem_bram acționează ca un buffer pentru stocarea temporară a imaginii capturate.
- **Furnizarea Datelor către vga_top**
  - **Intrări**:
    - **clk25m**: Acesta este clock-ul de 25 MHz utilizat pentru citirea datelor din memorie și furnizarea acestora către modulul VGA.
  - **Ieșiri**:

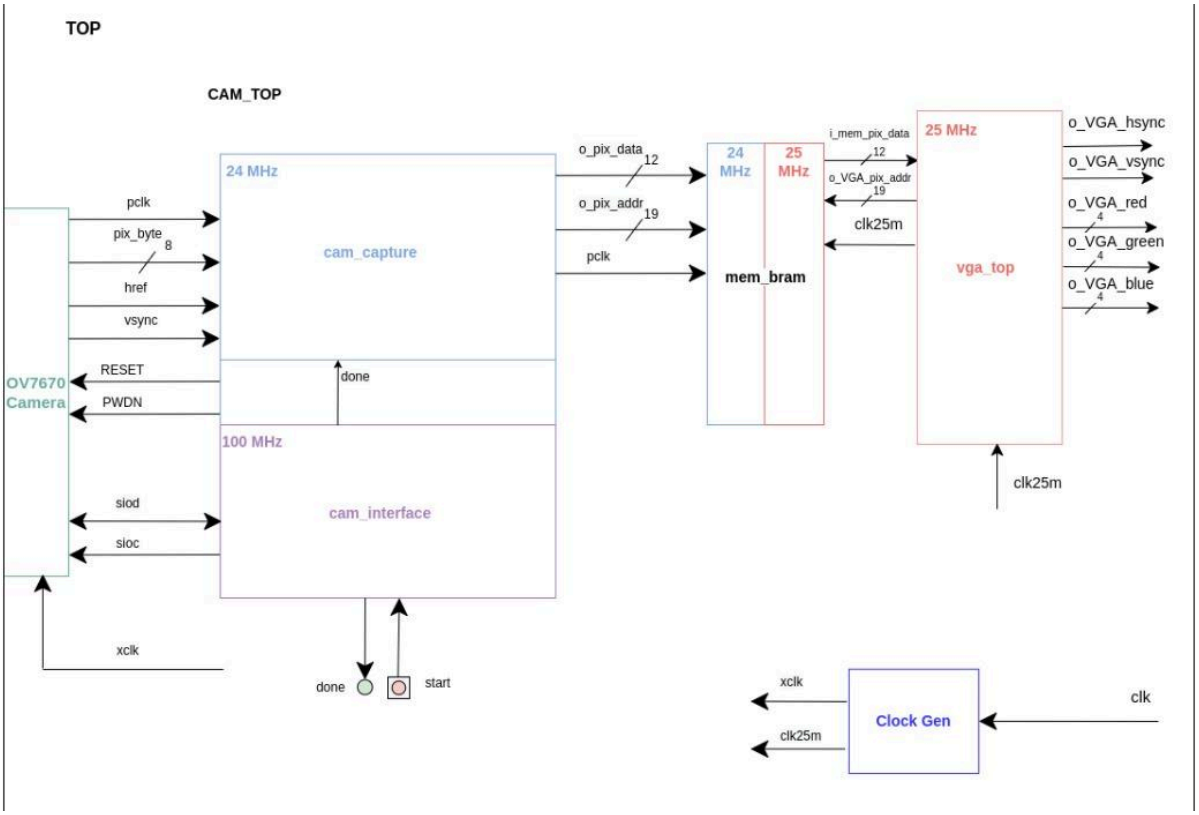
        - **i_mem_pix_data[11:0]**: Aceasta este magistrala de date pe 12 biți care furnizează datele pixelilor către modulul VGA.
        - **o_VGA_pix_addr[18:0]**: Aceasta este adresa pe 19 biți utilizată de modulul VGA pentru a citi datele corecte din memorie.
    - **Funcționalitate**:
        - În timpul fiecărui ciclu de clock **clk25m**, mem_bram furnizează datele pixelului specificat de adresa **o_VGA_pix_addr** către modulul VGA prin magistrala **i_mem_pix_data**. Modulul VGA utilizează aceste date pentru a genera semnalele necesare afișării imaginii pe display-ul VGA.

Rezumatul Modulului mem_bram

- **Scrierea Datelor**: Datele pixelilor capturate de camera OV7670 sunt stocate în memorie sincronizat cu clock-ul de pixeli (pclk). Adresa de memorie (o_pix_addr) specifică locația unde fiecare pixel este stocat.
- **Citirea Datelor**: Pentru afișare, datele sunt citite din memorie sincronizat cu clock-ul de 25 MHz (clk25m). Adresa de citire (o_VGA_pix_addr) este generată de modulul VGA pentru a accesa pixelii corecți din memorie.

Acest mecanism permite decuplarea temporară a procesului de capturare a imaginii de la camera și procesul de afișare pe display-ul VGA, permițând astfel un flux de date continuu și eficient între captură și afișare.

CAM TOP



Acest block diagram reprezintă un sistem care captează și afișează imagini de la o cameră OV7670 utilizând un display VGA. Diagrama este împărțită în mai multe module care colaborează pentru a realiza această funcție. Să le analizăm pe rând:

1. OV7670 Camera

- Senzorul OV7670: Acesta este senzorul de cameră care captează imaginile. Interfața sa include mai multe semnale:
    - pclk: Pixclock, semnal de clock pentru pixeli.
    - pix_byte[7:0]: Datele de pixeli.
    - href: Semnal de referință orizontală, indică linia activă a imaginii.
    - vsync: Semnal de sincronizare verticală, indică începutul unui nou cadru.
    - RESET: Semnal de resetare.
    - PWDN: Semnal de power down (oprire).
    - sioc: Clock pentru interfața de configurare I2C.

- siod: Date pentru interfața de configurare I2C.
- xclk: Clock extern pentru camera.

2. CAM_TOP Module

Acest modul este responsabil pentru interfațarea cu camera și capturarea datelor. Este împărțit în două submodule:

a. cam_capture

- cam_capture: Funcționează la 24 MHz și este responsabil pentru capturarea datelor de la cameră. Primeste semnale de la camera și trimite datele capturate la memorie (mem_bram).
  - Intrări: pclk, pix_byte[7:0], href, vsync.
  - Iesiri: o_pix_data[11:0] (datele pixelilor capturate), o_pix_addr[18:0] (adresa pixelilor în memorie).
  - Semnal de sincronizare: done (indică faptul că datele au fost capturate).

b. cam_interface

- cam_interface: Funcționează la 100 MHz și este responsabil pentru configurarea camerei prin interfața I2C.
  - Intrări: start (semnal de început).
  - Ieșiri: done (semnal care indică terminarea configurării).

3. mem_bram Module

- mem_bram: Acest modul servește ca o memorie intermediară (Block RAM) pentru stocarea datelor de pixeli capturate de la cameră și furnizarea acestora către modulul VGA.
  - Intrări: pclk (clock pentru scriere), clk25m (clock pentru citire), o_pix_data[11:0], o_pix_addr[18:0].
  - Ieșiri: i_mem_pix_data[11:0], o_VGA_pix_addr[18:0].

4. vga_top Module

- vga_top: Funcționează la 25 MHz și este responsabil pentru generarea semnalelor VGA pentru afișarea imaginii.
    - Intrări: clk25m, i_mem_pix_data[11:0], o_VGA_pix_addr[18:0].
    - Ieșiri: o_VGA_hsync (semnal de sincronizare orizontală), o_VGA_vsync (semnal de sincronizare verticală), o_VGA_red[3:0], o_VGA_green[3:0], o_VGA_blue[3:0] (semnale RGB pentru culori).

5. Clock Gen Module

- Clock Gen: Acest modul generează diverse semnale de clock necesare pentru sistem, inclusiv xclk pentru camera și clk25m pentru memorie și VGA.
    - Intrări: clk (clock principal).
    - Ieșiri: xclk (clock pentru camera), clk25m (clock pentru memorie și VGA).

Fluxul de date și control

1. Capturarea imaginilor: Modulul cam_capture preia datele de pixeli de la camera OV7670 și le stochează în memorie (mem_bram).
2. Stocarea în memorie: Datele de pixeli sunt stocate în mem_bram care funcționează ca o memorie intermediară.
3. Afișarea pe VGA: Modulul vga_top preia datele de pixeli din memorie și generează semnalele necesare pentru afișarea imaginii pe un display VGA.

Acesta este un sistem tipic pentru preluarea și afișarea imaginilor de la o cameră digitală folosind FPGA și este util pentru aplicații de vizualizare în timp real.