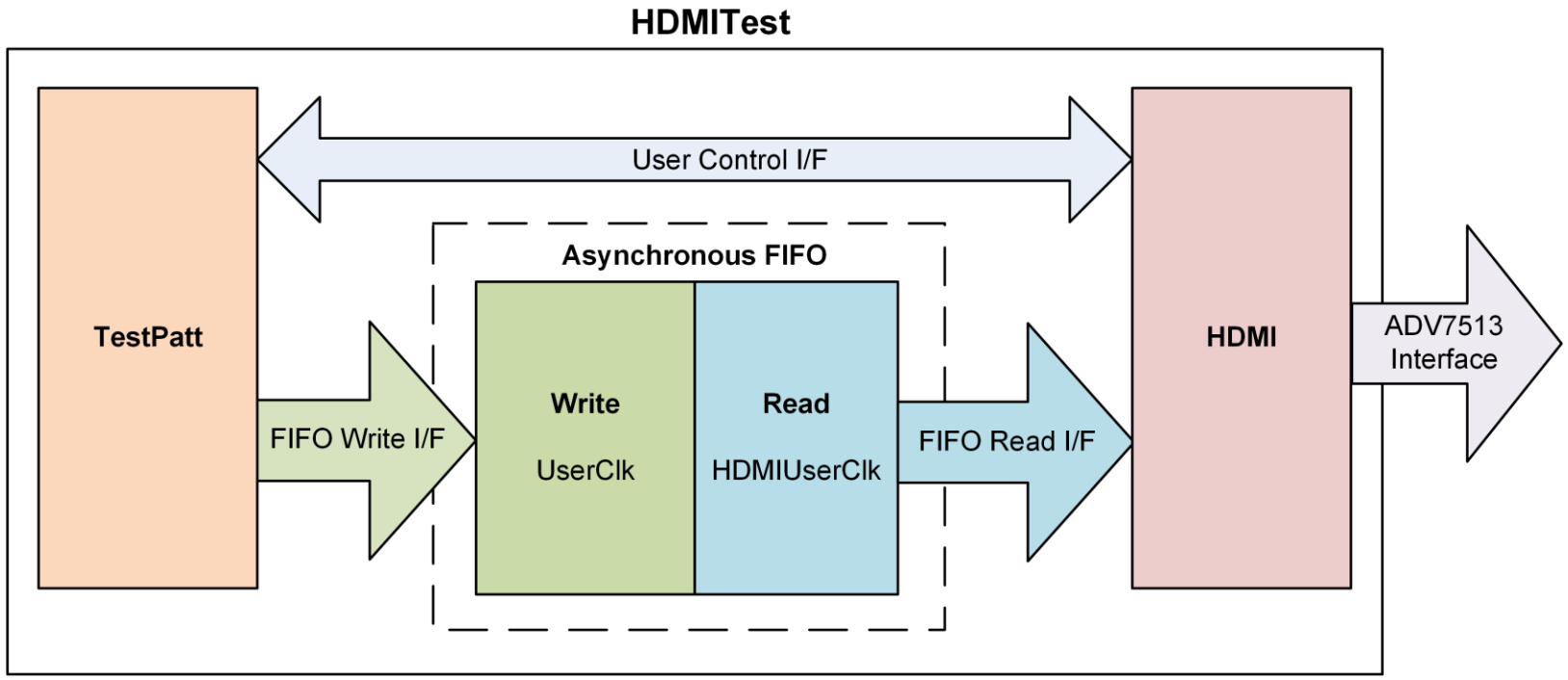
DIGITAL DESIGN WITH FPGA CAMP

CONTEST 2 : BITMAP PATTERN TO HDMI WITH DDR BUFFER

ELECTRICAL ENGINEERING
CHULALONGKORN UNIVERSITY
TESA
NDR Solution
DG DESIGN GATEWAY
Silicon craft

HDMITEST

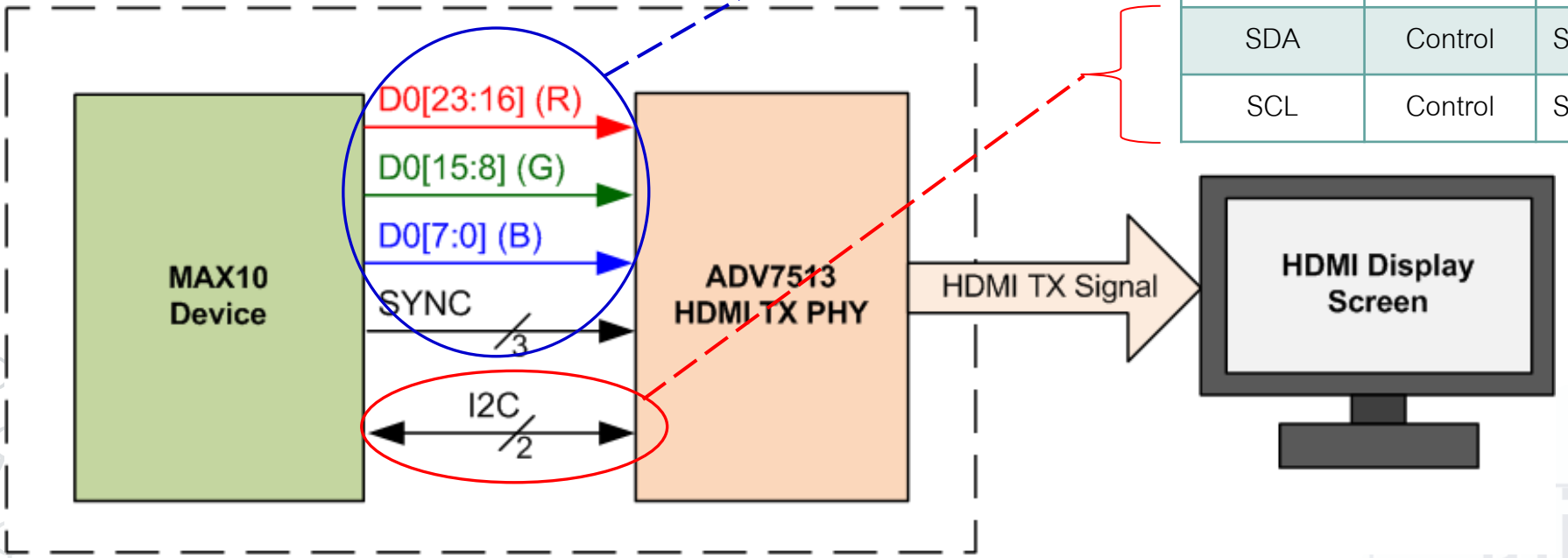
HDMITEST BLOCK DIAGRAM



- วงจรตัวอย่างที่จะส่งข้อมูลภาพทดสอบเป็น Color bar แนวตั้งไปยังจอผ่าน HDMI interface ด้วย resolution 1024x768 pixel
- ข้อมูลระหว่างวงจรสร้างภาพ (TestPatt) กับวงจรแสดงภาพ (HDMI) จะถูกส่งผ่าน FIFO
- วงจร TestPatt ทำงานภายใต้ Clk ที่ชื่อ UserClk ในขณะที่วงจร HDMI ทำงานภายใต้ Clk ที่ชื่อ HDMIUserClk

MAX10 <-> HDMI TX INTERFACE

Mnemonic	Type	Description
D[23:0]	Input	Video Data Inputs.
VSYNC	Input	Vertical Synchronization Input.
HSYNC	Input	Horizontal Synchronization Input.
DE	Input	Data Enable Signal for Digital Video.
CLK	Input	Video Input Clock.
INT	Output	Interrupt Signal Output.
SDA	Control	Serial Port Data Input/Output.
SCL	Control	Serial Port Data Clock Input.

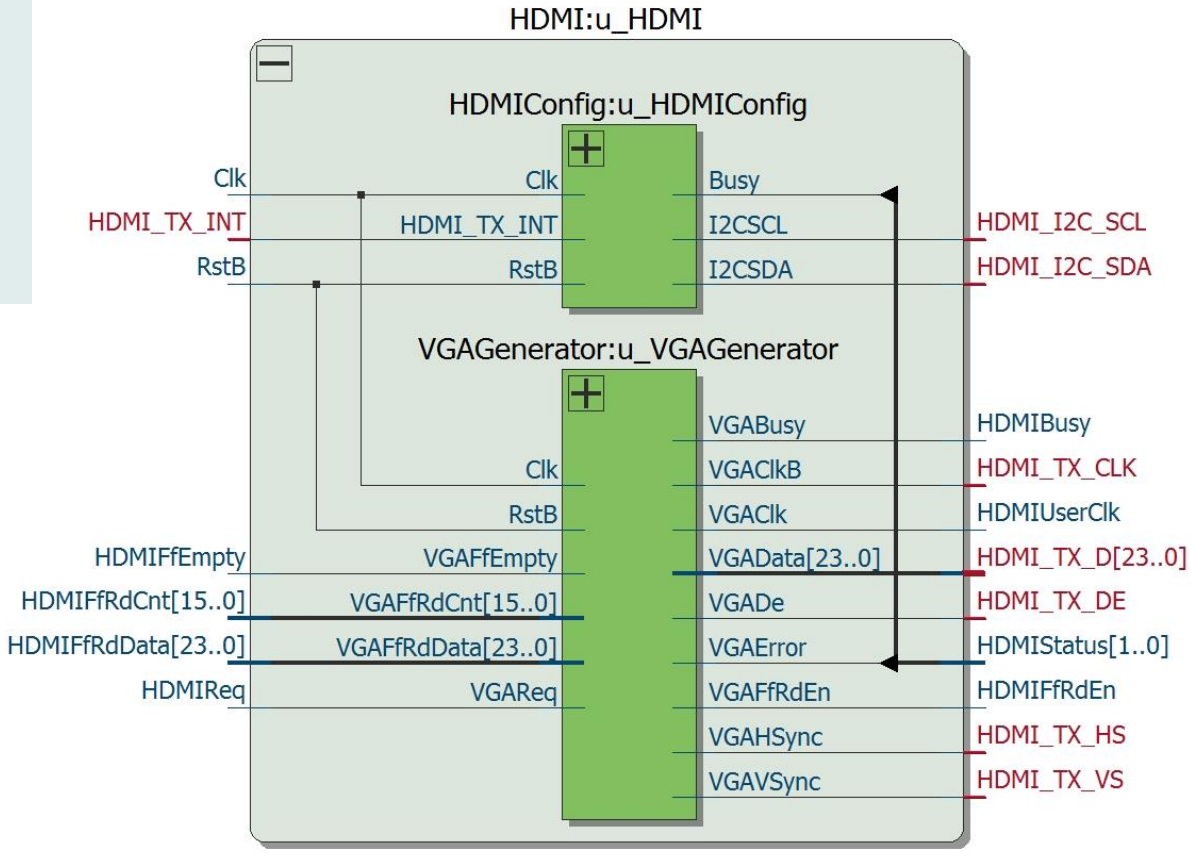


HDMI MODULE

- HDMI เป็น IP ที่สร้างขึ้นสำหรับการ Interface ระหว่าง Max10 กับ AD7513
- สัญญาณสีแดง คือ สัญญาณที่ขาของ FPGA สำหรับ Interface กับ AD7513
- สัญญาณชื่อสีน้ำเงินคือสัญญาณภายในสำหรับผู้ใช้งานใช้ควบคุมการทำงานของ HDMI

ภายใน HDMI-IP จะถูกแบ่งออกเป็น 2 ส่วน

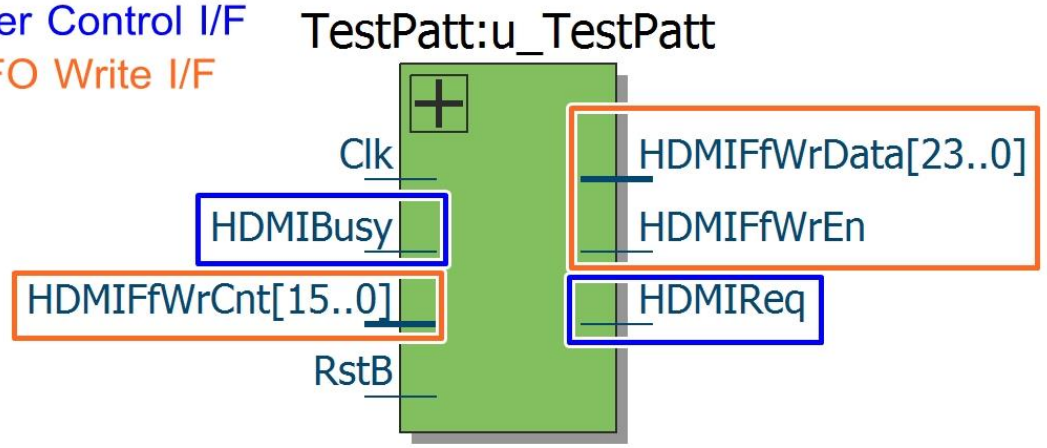
1) HDMIConfig ใช้สร้างสัญญาณควบคุมการทำงานด้วย I2C Protocol
2) VGAGenerator ใช้สร้างสัญญาณข้อมูลสำหรับการแสดงผลบนจอภาพด้วย VGA Protocol



TESTPATT

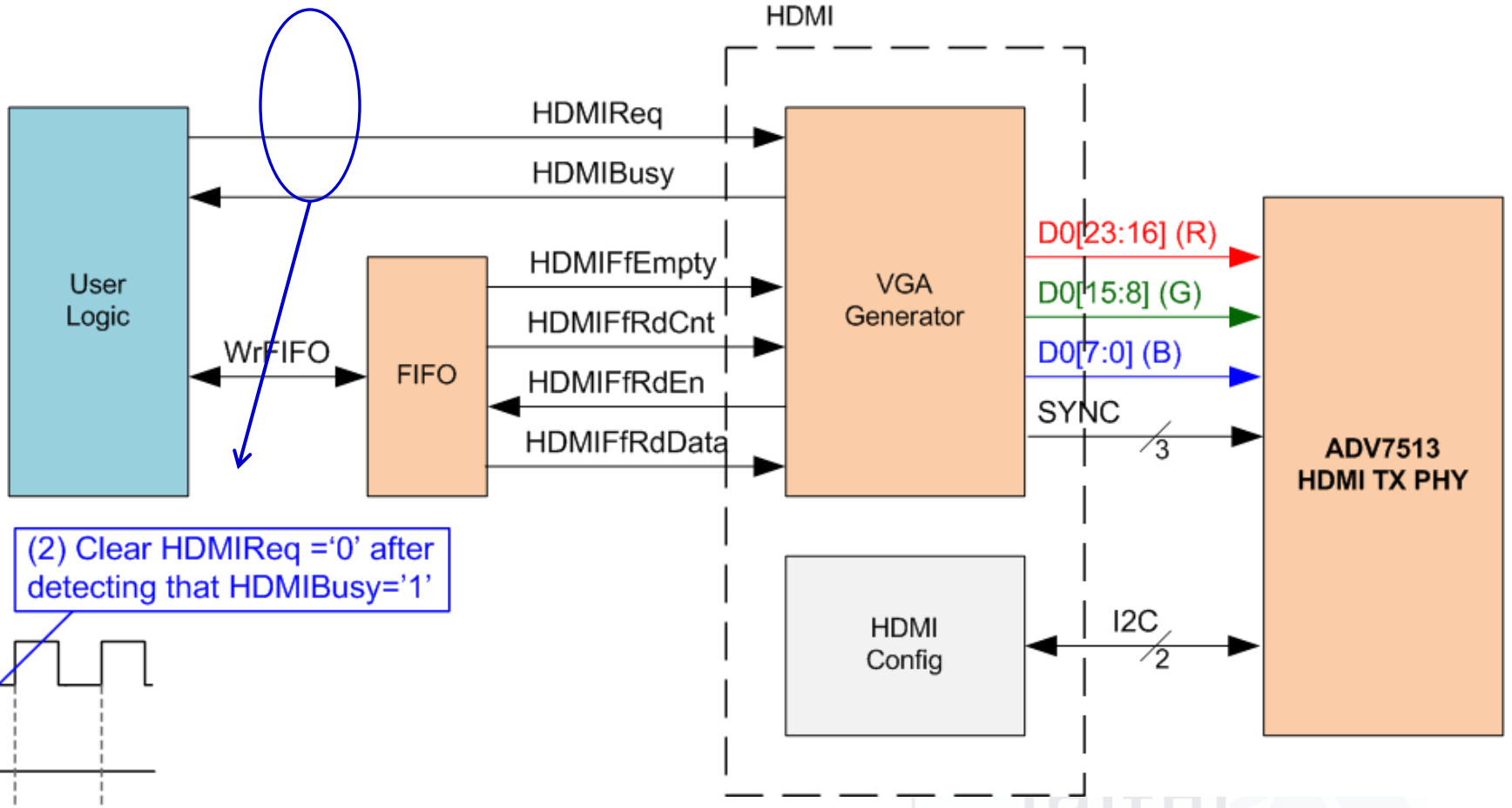
- สร้าง HDMIReq เพื่อให้ HDMI module เริ่มทำงาน
- เขียนข้อมูลลง FIFO ซึ่งเป็นข้อมูล Color Bar pattern เพื่อแสดงภาพ

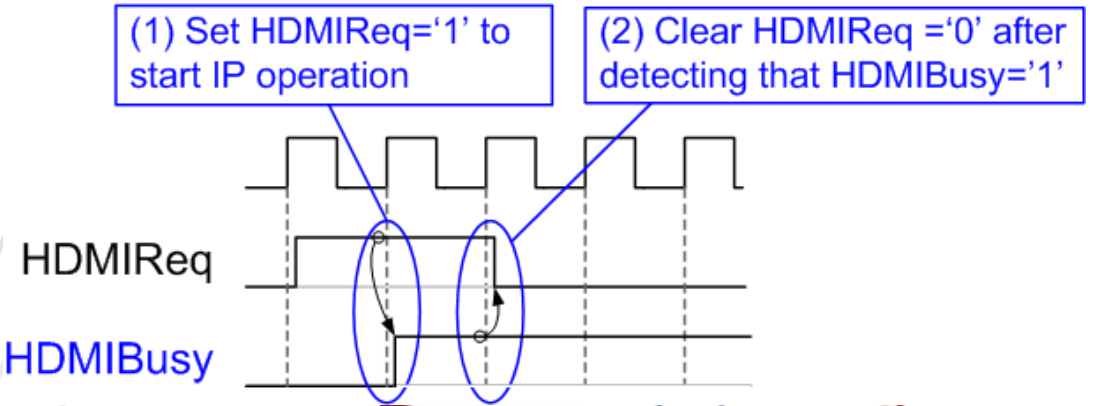
Blue : User Control I/F
Orange : FIFO Write I/F



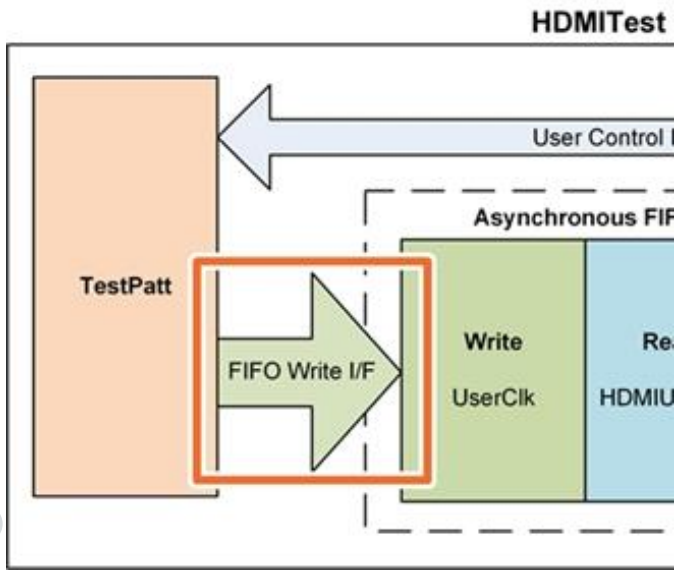
HDMI USER INTERFACE

- ส่งสัญญาณ HDMIReq ไปเพื่อให้เริ่มทำงานเพียงครั้งเดียว HDMI module จะรอจนถึงจุดเริ่มต้นของภาพ แล้วจึงเริ่มอ่านข้อมูลจาก FIFO ส่งไปที่ HDMI TX PHY ตลอดเวลา
- Error (HDMIStatus[1]) จะมีค่าเป็น '1' เมื่อเจอ FIFOEmpty='1' ระหว่างส่งข้อมูลภาพ





HDMITEST : FIFO WRITE I/F



TestPatt มีหน้าที่ในการสร้างข้อมูลที่จะแสดงออกบนจอภาพแล้วเขียนลงไปใน FIFO บน clock domain ชื่อ UserClk

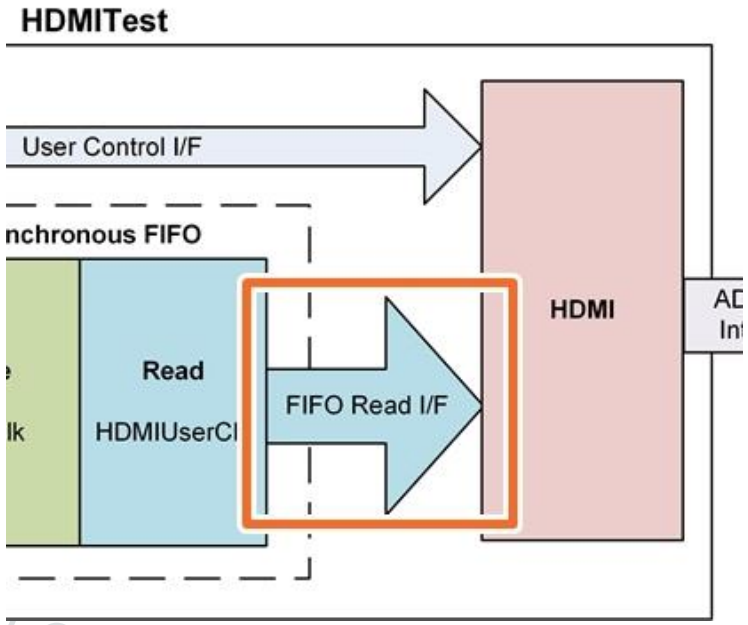
```
241   -- HDMI test pattern generator
242   u_TestPatt : TestPatt
243   Port map
244   (
245       RstB            => rSysRstB
246       Clk             => UserClk
247
248       HDMIReq         => HDMIReq
249       HDMIBusy        => HDMIBusy     ,
250
251       HDMIFfWrEn      => HDMIFfWrEn   ,
252       HDMIFfWrData    => HDMIFfWrData ,
253       HDMIFfWrCnt     => HDMIFfWrCnt
254   );

      -- Fill zeros
      HDMIFfWrCnt(15 downto 8)    <= (others=>'1');
      HDMIFfRdCnt(15 downto 8)    <= (others=>'0');

      -- AsyncFifo TestPatt -> HDMI
      u_asyncfifo256x24 : asyncfifo256x24
      Port map
      (
          aclr            => SysRst       ,
265
266       wrclk           => UserClk      ,
267       wrreq           => HDMIFfWrEn   ,
268       data            => HDMIFfWrData ,
269       wrfull          => open         ,
270       wrusedw         => HDMIFfWrCnt(7 downto 0),
271
272       rdclk           => HDMIUserClk  ,
273       rdreq           => HDMIFfRdEn   ,
274       q               => HDMIFfRdData ,
275       rdempty         => HDMIFfEmpty  ,
276       rdusedw         => HDMIFfRdCnt(7 downto 0)
277   );
```

เติมค่า HDMIFfWrCnt ให้มีขนาด 16 บิตด้วยการใส่ 1 ที่ MSB เพื่อเช็คพื้นที่ว่างใน FIFO

HDMITEST : FIFO READ I/F



เติมค่า HDMIFfRdCnt ให้มีขนาด 16 บิตด้วยการใส่ 0 ล้วนเพื่อเช็คจำนวนข้อมูลใน FIFO ที่มีอยู่

HDMI จะอ่านข้อมูลจาก FIFO มาแสดงผลที่จอภาพบน clock domain ชื่อ HDMIUserClk

```
256   -- Fill zeros
257   HDMIFfWrCnt(15 downto 8)    <= (others=>'1');
258   HDMIFfRdCnt(15 downto 8)    <= (others=>'0');
259
260   -- AsyncFifo TestPatt -> HDMI
261   u_asyncfifo256x24 : asyncfifo256x24
262   Port map
263   (
64        aclr            => SysRst           ,
65
66        wrclk           => UserClk          ,
67        wrreq           => HDMIFfWrEn       ,
68        data            => HDMIFfWrData     ,
69        wrfull          => open             ,
70        wrusedw         => HDMIFfWrCnt(7 downto 0),
71
72        rdclk           => HDMIUserClk      ,
273       rdreq           => HDMIFfRdEn       ,
274       q               => HDMIFfRdData     ,
275       rdempty         => HDMIFfEmpty      ,
276       rdusedw         => HDMIFfRdCnt(7 downto 0)
277   );
278
279   -- HDMI Display interface
280   u_HDMI : HDMI
281   Port map
282   (
283       RstB            => rSysRstB         ,
284       Clk             => UserClk          ,
285
286       HDMIReq         => HDMIReq          ,
287       HDMIBusy        => HDMIBusy         ,
288       HDMIStatus      => HDMIStatus       ,
289       HDMIUserClk     => HDMIUserClk      ,
290
291       HDMIFfRdEn      => HDMIFfRdEn       ,
292       HDMIFfRdData    => HDMIFfRdData     ,
293       HDMIFfEmpty     => HDMIFfEmpty      ,
294       HDMIFfRdCnt     => HDMIFfRdCnt      ,
295
296       HDMI_TX_INT     => HDMI_TX_INT      ,
297       HDMI_I2C_SCL    => HDMI_I2C_SCL     ,
298       HDMI_I2C_SDA    => HDMI_I2C_SDA     ,
299
300       HDMI_TX_CLK     => HDMI_TX_CLK      ,
301       HDMI_TX_D       => HDMI_TX_D        ,
302       HDMI_TX_DE      => HDMI_TX_DE       ,
303       HDMI_TX_HS      => HDMI_TX_HS       ,
304       HDMI_TX_VS      => HDMI_TX_VS
305   );
```

TESTPATT : WRITE FIFO ENABLE

```
u_rHDMIFfWrEn : Process (Clk) Is
Begin
    if ( rising_edge(Clk) ) then
        if ( RstB='0' ) then
            rHDMIFfWrEn(2 downto 0)     <= "000";
        else
            rHDMIFfWrEn(2 downto 1) <= rHDMIFfWrEn(1 downto 0);
            -- Break when free space is less than 8
            if ( HDMIFfWrCnt(15 downto 3)/=('1'&x"FFF") ) then
                rHDMIFfWrEn(0)  <= '1';
            else
                rHDMIFfWrEn(0)  <= '0';
            end if;
        end if;
    end if;
End Process u_rHDMIFfWrEn;
```

เช็คที่ว่างใน FIFO หากมีที่ว่างไม่น้อยกว่า 8 จะเขียนข้อมูลถัดไปลงไปใน FIFO (ใช้ 8 เผื่อไว้เป็น latency ของกระบวนการ process สัญญาณ)

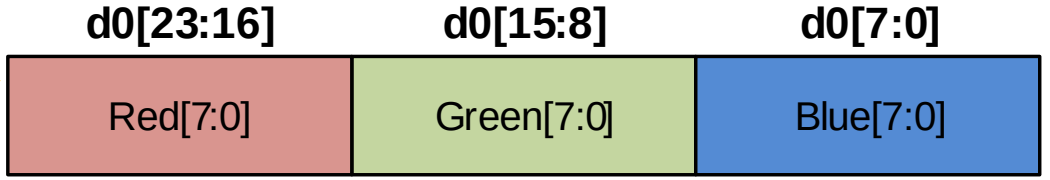
```
-------------------------------------------------------------------
-- Output assignment
-------------------------------------------------------------------

    HDMIReq                     <= rHDMIReq;

    HDMIFfWrEn                  <= rHDMIFfWrEn(2);
    HDMIFfWrData(7 downto 0)    <= rHDMIBlue(7 downto 0);
    HDMIFfWrData(15 downto 8)   <= rHDMIGreen(7 downto 0);
    HDMIFfWrData(23 downto 16)  <= rHDMIRed(7 downto 0);
```

- rHDMIFfWrEn(0) : สัญญาณสำหรับนับ rHCnt และ rVCnt ซึ่งจะนำไปเป็น input สำหรับสร้าง HDMIFfWrData
- สัญญาณ rHDMIBlue/Green/Red จะพร้อมใช้งานหลัง rHDMIFfWrEn(0)='1' ไป 2 clock
- HDMIFFWrEn ถูกสร้างจาก rHDMIFfWrEn(2) เพื่อ sync กับสัญญาณ HDMIFfWrData

DISPLAY SEQUENCE TO WRITE FIFO (HDMIFFWRDATA)



x0 → x1023

y0 ↓ y767

d0	d1	d2	....	d1021	d1022	d1023
d1024	d1025	d1026	....	d2045	d2046	d2047
....	....	....	....	....	....	...
d784384	d784385	d784386		d785405	d758406	d785407
d785408	d785409	d785410	....	d786429	d786430	d786431

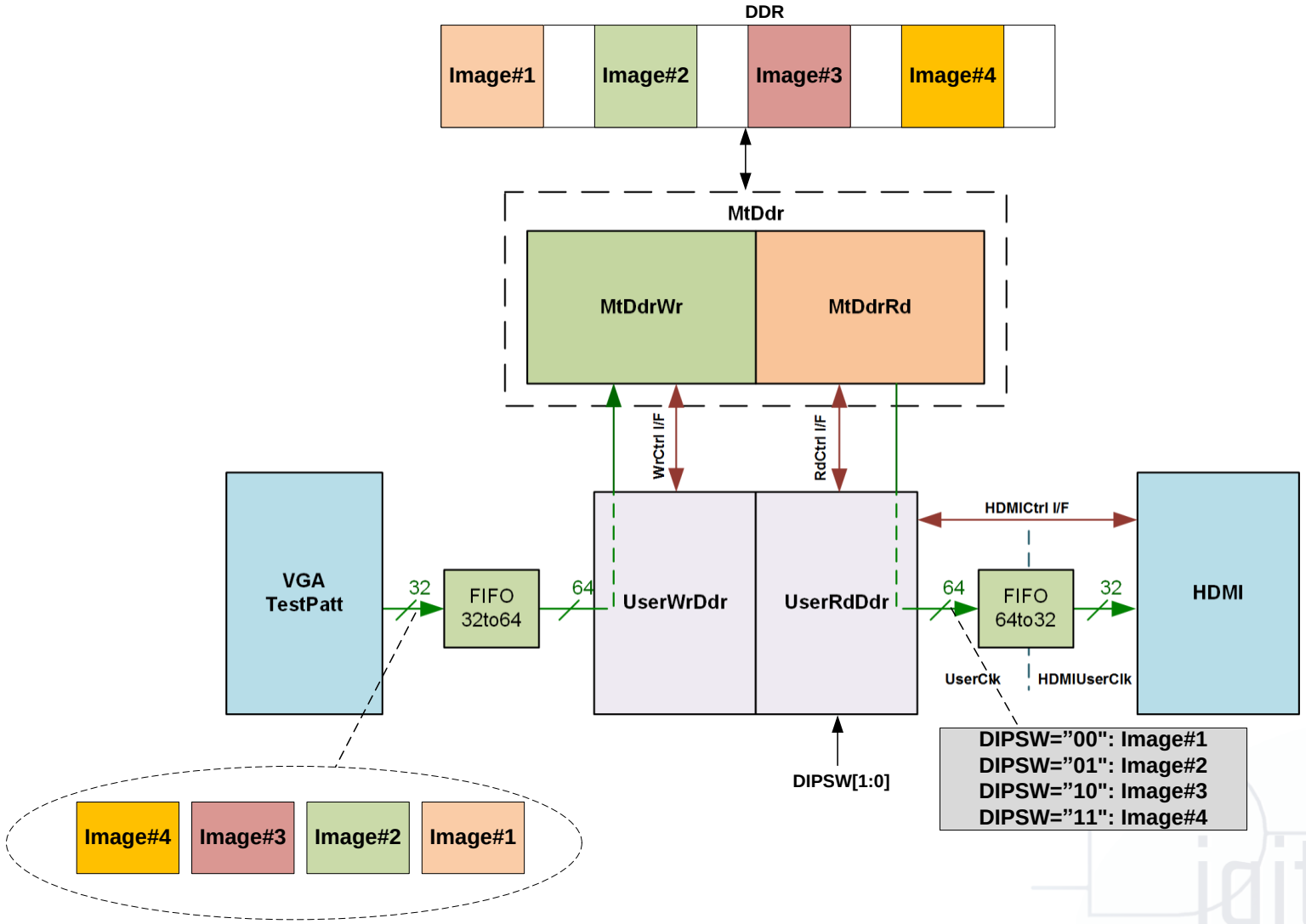
ข้อมูลแต่ละ Pixel ประกอบด้วยค่า RGB อย่างละ 8 บิตเพื่อบอกความสว่างของ Red/Green/Blue

ข้อมูลที่แสดงออกจอภาพที่มีขนาด frame คงที่ 1024x768

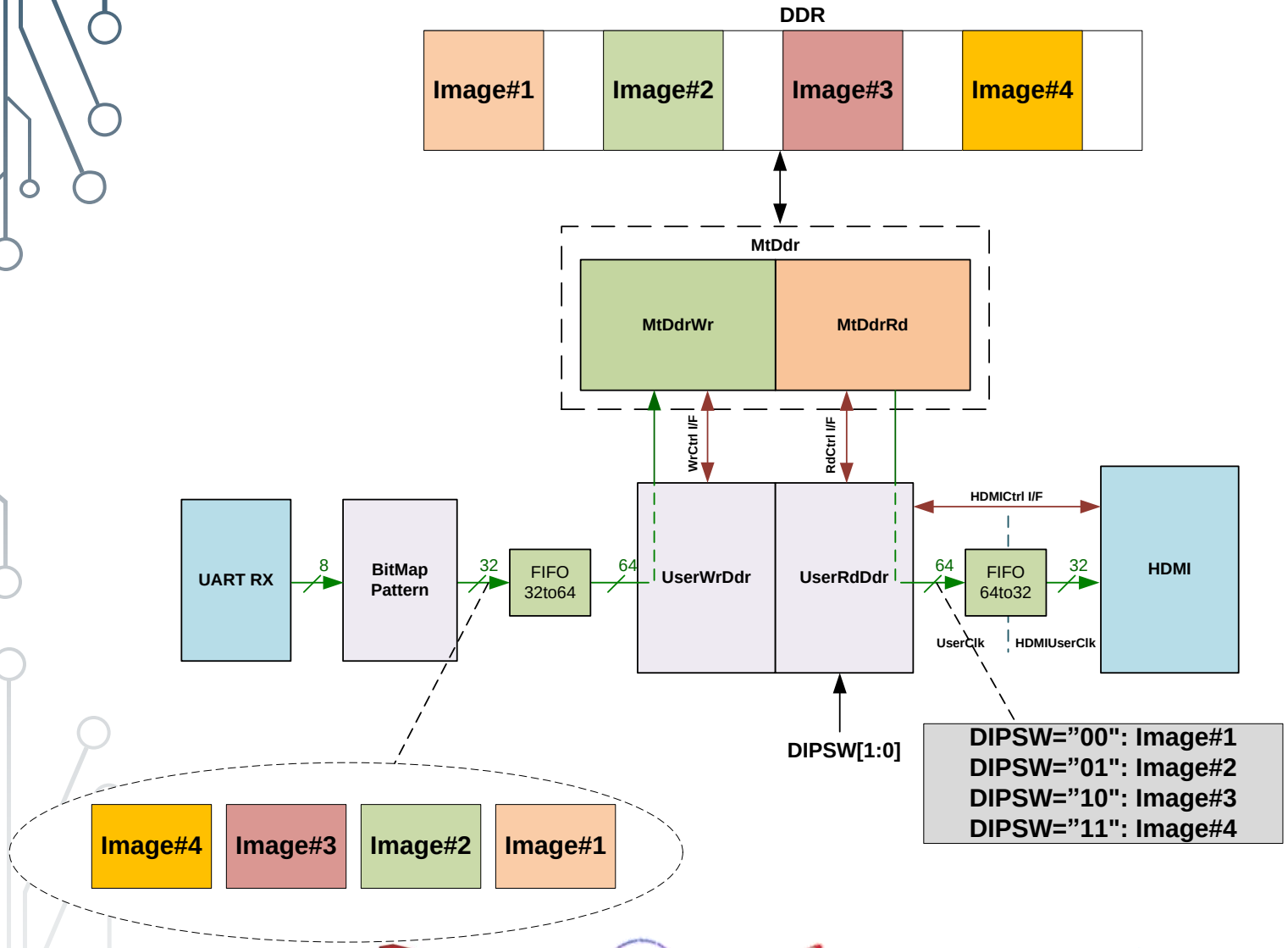
ลำดับการเขียน Data ลง FIFO จะเริ่มจาก d0 ถึง d786431 เมื่อครบภาพแล้ว จะกลับมาส่งที่ d0 ใหม่

CONTEST2

CONTEST 1 TEST PATTERN TO HDMI WITH DDR BUFFER

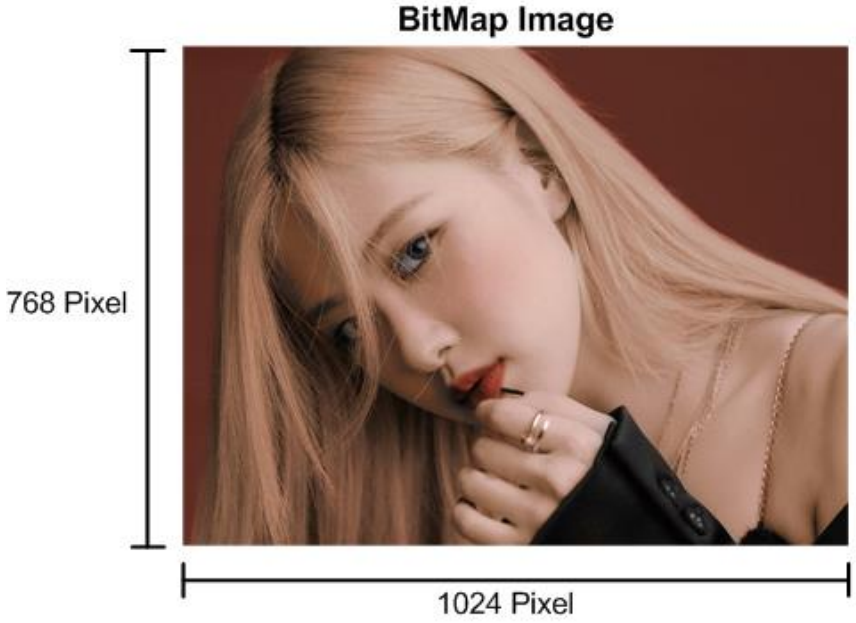


CONTEST 2 BITMAP PATTERN TO HDMI WITH DDR BUFFER

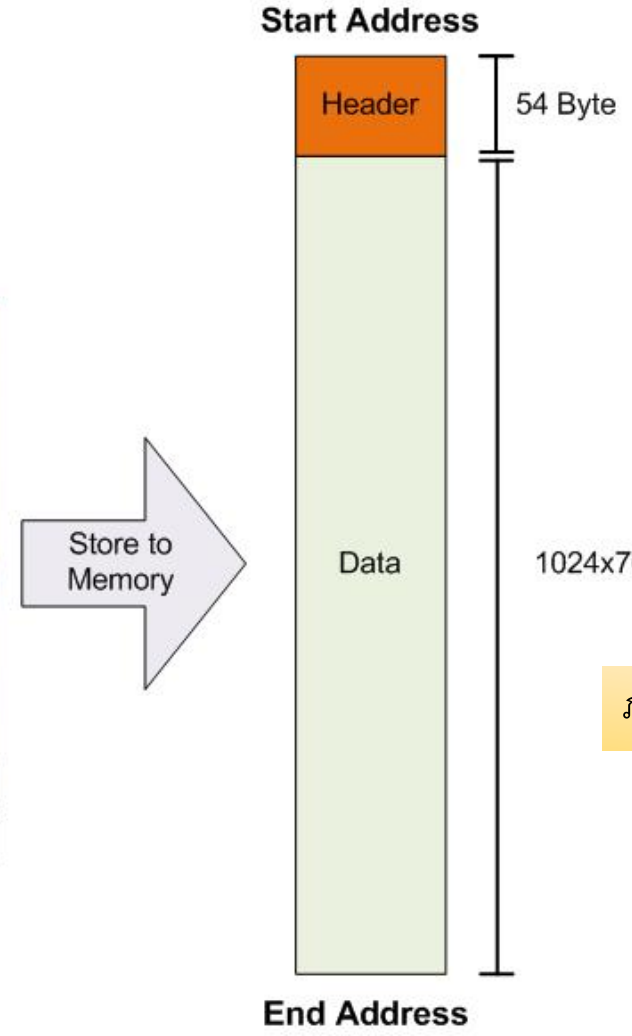


1. เปลี่ยนจาก VGATestPatt เป็นวงจร UART RX เพื่อรับข้อมูลภาพมาจาก TeraTerm แทน
2. เพิ่มวงจร BitMap pattern เพื่อนำเฉพาะข้อมูลที่เป็นภาพจากไฟล์ มาเก็บไว้ใน FIFO เท่านั้น
3. แก้ไขวงจร UserWrDdr ในส่วนที่ควบคุมสัญญาณ Address ให้มีลำดับการส่ง Address ให้ตรงกับลำดับของข้อมูล Bitmap
4. เมื่อได้ข้อมูลครบ 1 ภาพ หากมีข้อมูลมาใหม่ ก็จะปรับตำแหน่งไปเริ่มต้นที่ Address แรกของ Slot 128 Mbyte ถัดไป (Address[28:27] เพิ่มขึ้น 1 และ Address[26:0]=0)

BITMAP FILE FORMAT

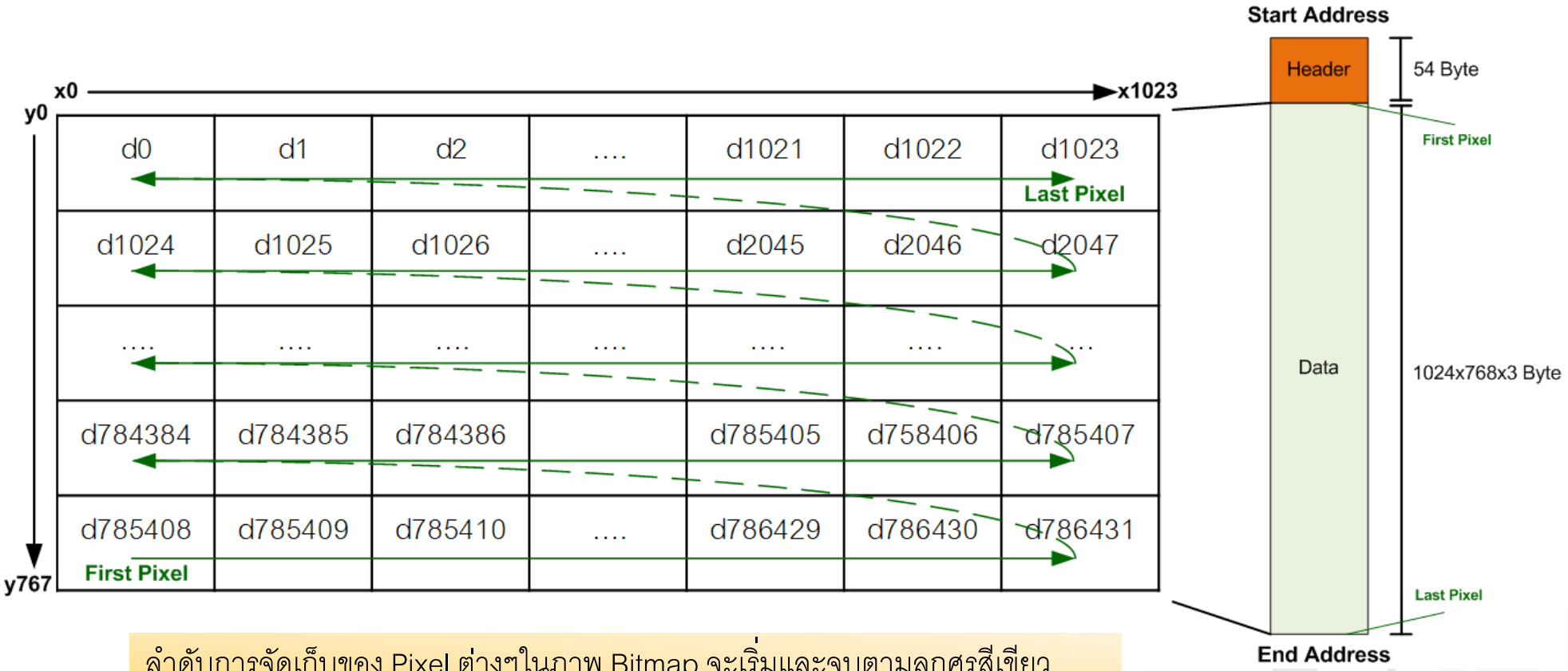


ภาพ Bitmap ที่แสดงผล



ภาพ Bitmap ที่ถูกเก็บไว้ในหน่วยความจำ

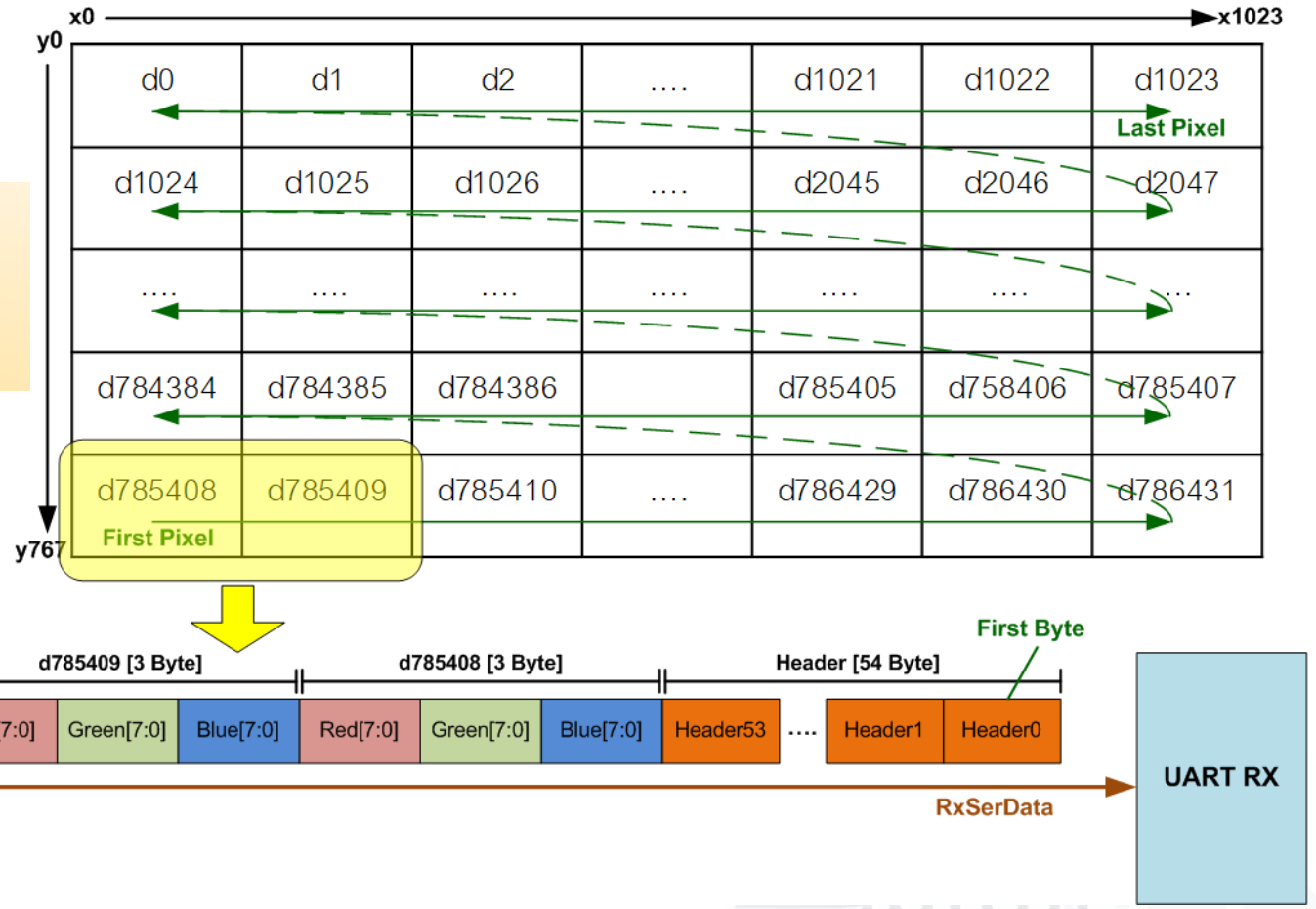
BITMAP FILE FORMAT



ลำดับการจัดเก็บของ Pixel ต่างๆในภาพ Bitmap จะเริ่มและจบตามลูกศรสีเขียว

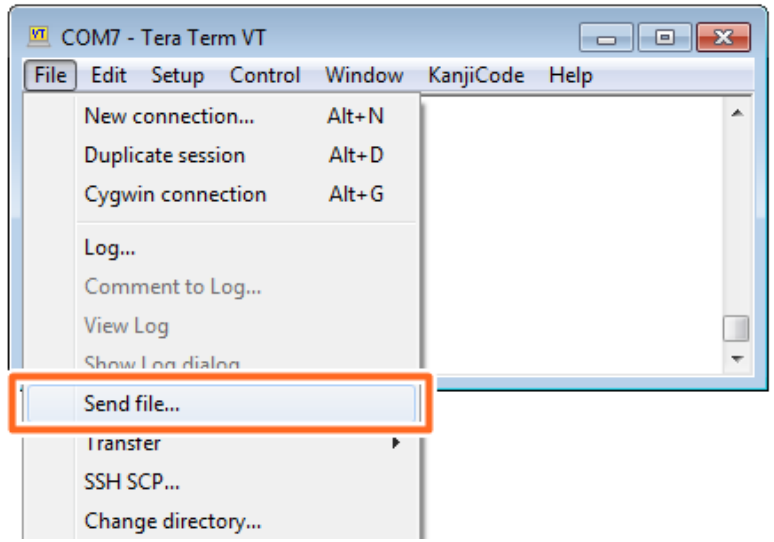
SEND BITMAP WITH UART

คอมพิวเตอร์จะส่งข้อมูลภาพ Bitmap โดยมีลำดับการส่งคือ จาก Start Address ของ Bitmap ไปสู่ End Address (ส่ง Header 54 Byte แล้วจึงส่ง d785408 เป็นข้อมูลตัวแรก)



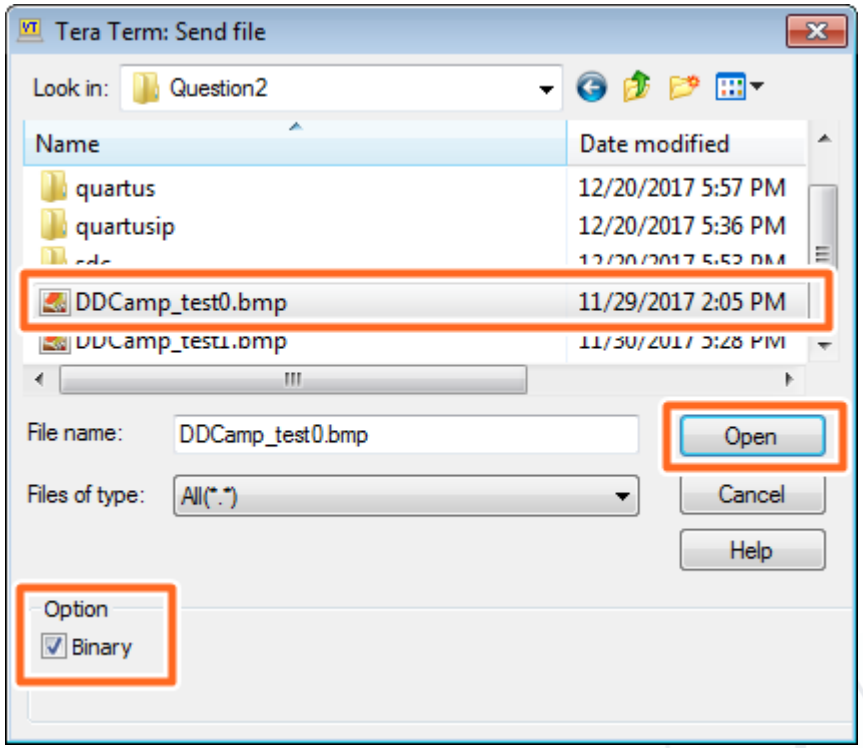
SEND BITMAP WITH TERA TERM

1) เปิด Tera Term ตั้งค่า Baud Rate ให้เรียบร้อยแล้วเลือก File -> Send file…



SEND BITMAP WITH TERA TERM

2) เลือก File ภาพ Bitmap ที่ต้องการ

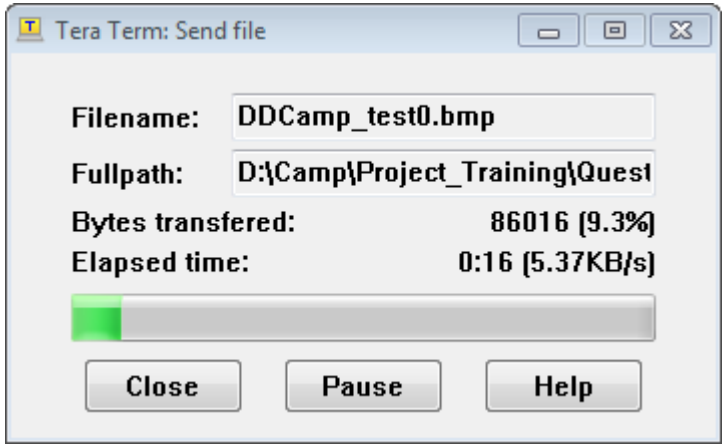


3) ติ๊ก Binary Option ให้เป็นเครื่องหมายถูกดังภาพ

4) เลือก Open

SEND BITMAP WITH TERA TERM

5) รอจนหน้าต่างนี้หายไปการส่งไฟล์ถึงเสร็จสิ้น



หมายเหตุ : สถานะในการส่งไฟล์นี้จะระบุไม่ตรงกับค่าสถานะที่โปรแกรมแสดง