

**FPGA 기반 프레임그래버를 이용한  
영상처리(NI 1473R)**

**2016. 9. 7.**

**한국생산기술연구원 이은주**

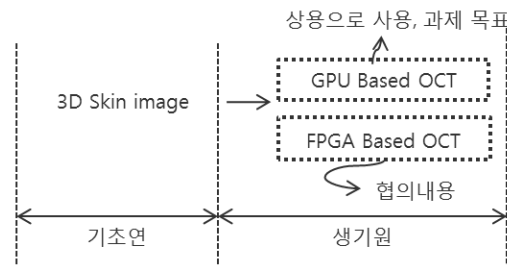
## 목 차

1. 개요-----	2
2. 시스템 설치-----	3
3. 시스템 Labview 파일 -----	6
A. Labview project 개요 -----	6
B. Labview project 설정 -----	11
C. Labview project 실행 -----	14
D. Labview project 실행 결과 -----	15
4. 영상처리 -----	16
5. 부록 -----	18

## 1. 개요

### A. 목표

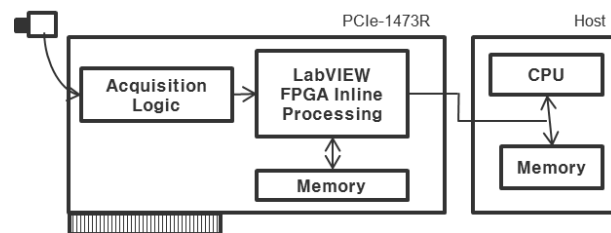
- FPGA 기반의 생체광학 3D 영상처리를 위하여 NI 사의 FPGA 기반 Frame grabber 보드인 PCIe-1473R 보드를 사용하였으며 이 보드를 이용하여 FPGA Based OCT 수행



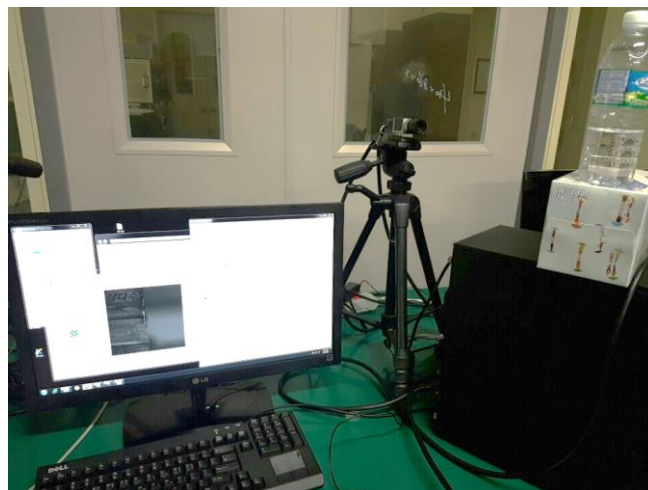
<그림 1 기관 협의 내용>

### B. 시스템 구성

- FPGA Based OCT 시스템은 카메라 PCIe-1473R, Host PC 로 구성



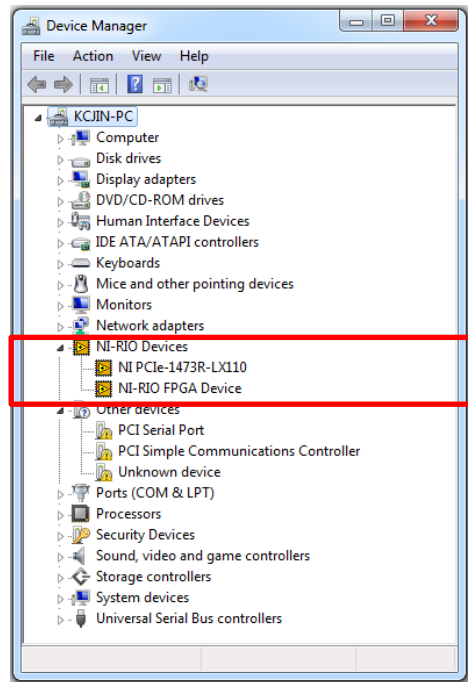
<그림 2. 시스템 구성 Block diagram>



<그림 3. 시스템 구성>

## 2. 시스템 설치

### A. PCIe-1473R 보드 Host PC 슬롯에 장착



<그림 4. 디바이스 매니저 화면>

### B. 랩뷰 설치 (CD)

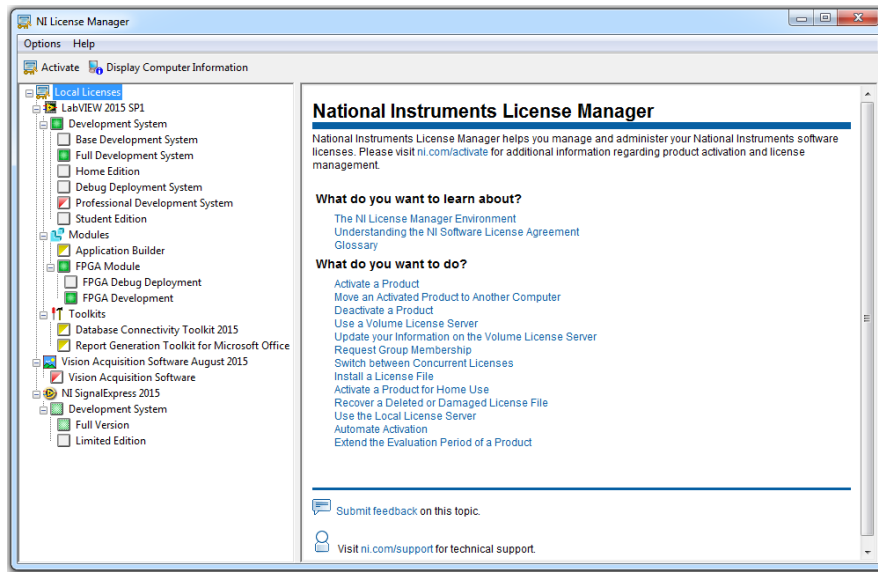
- 순서대로 설치, 정품인증

### C. 랩뷰 FPGA Module 설치(CD)

- 순서대로 설치, 정품인증
- Labview real-time. Labview FPGA, NI-1473R Driver 설치

### D. Ni 카메라 관련 (NI VISION Acquisition Software August 2015)

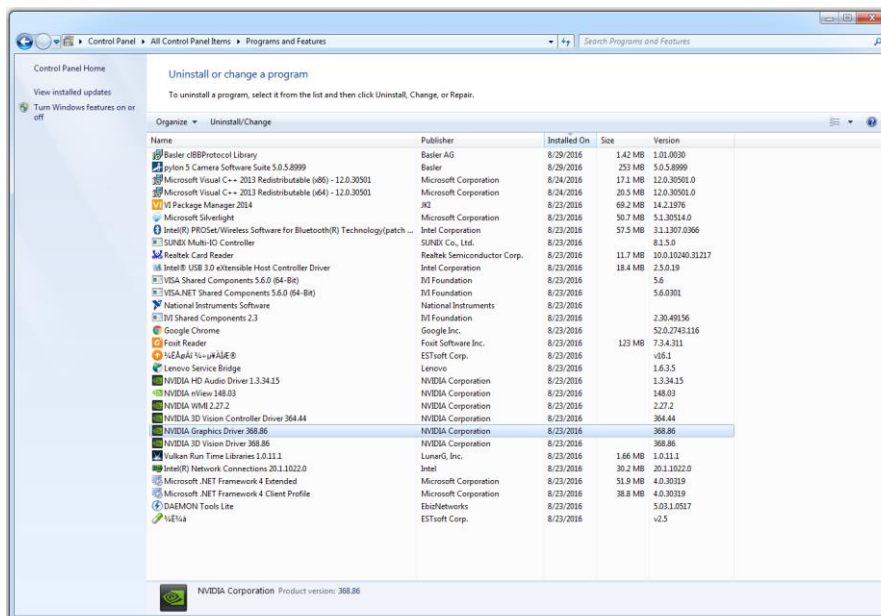
- 랩뷰 설치시 함께 설치 되지 않는다면 따로 설치



<그림 5. NI 라이선스 매니저 화면>

#### E. Ni FPGA Module Xilinx tool

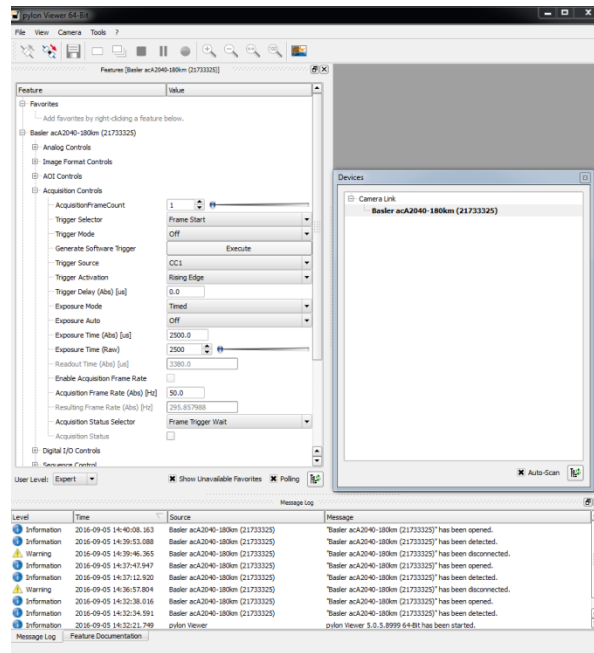
- 랩뷰 FPGA Module 설치시 함께 설치 되지 않는다면 따로 설치



<그림 6. 설치 파일 화면>

#### F. 카메라의 레지스트를 수정할 수 있는 프로그램 설치(카메라 브랜드마다 다름)

- Image Width, Height, Tap 정보 필수.

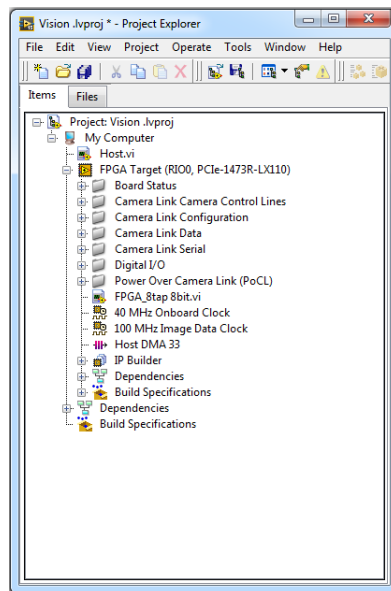


<그림 7. 카메라 설정 화면>

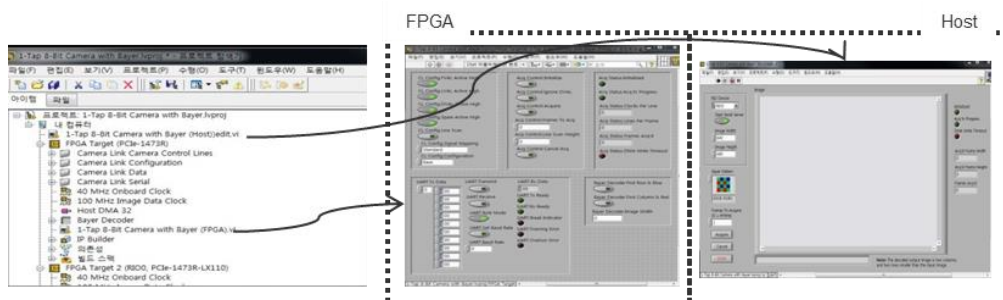
### 3. 시스템 Labview 파일

#### A. Labview project 개요

- .lvproj 파일을 실행하면 Host.vi(Host PC)와 FPGA\_8tap\_8bit.vi(FPGA) 파일로 구분됨
- 카메라로부터 영상을 입력 받고 DRAM 저장 없이 출력(디스플레이) 프로그램



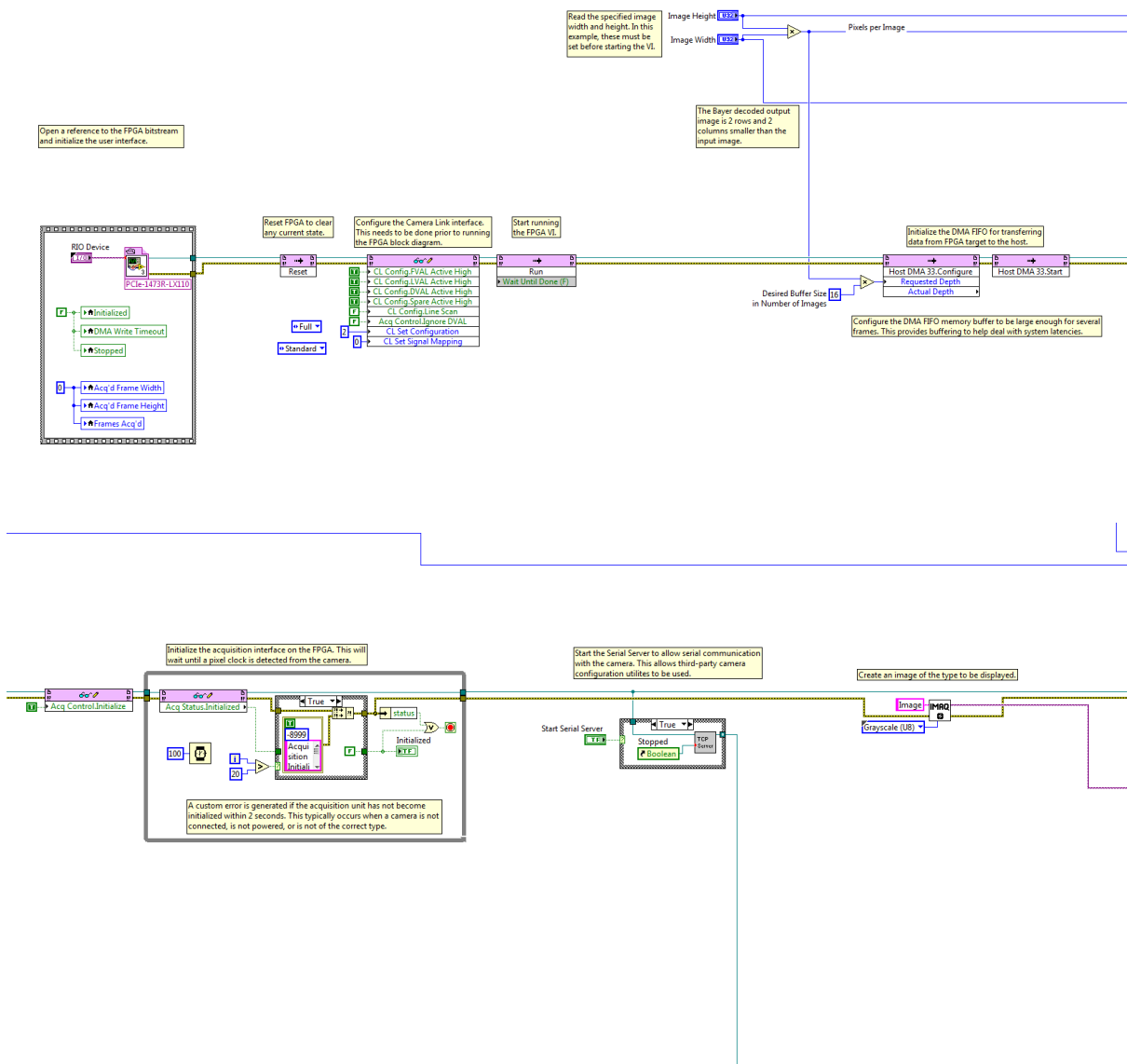
<그림 8. .lvproj 파일 화면>



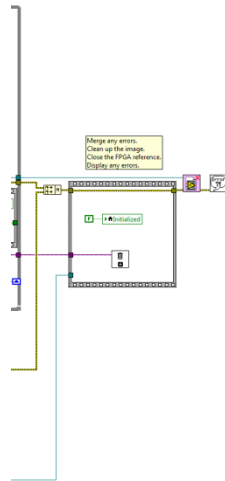
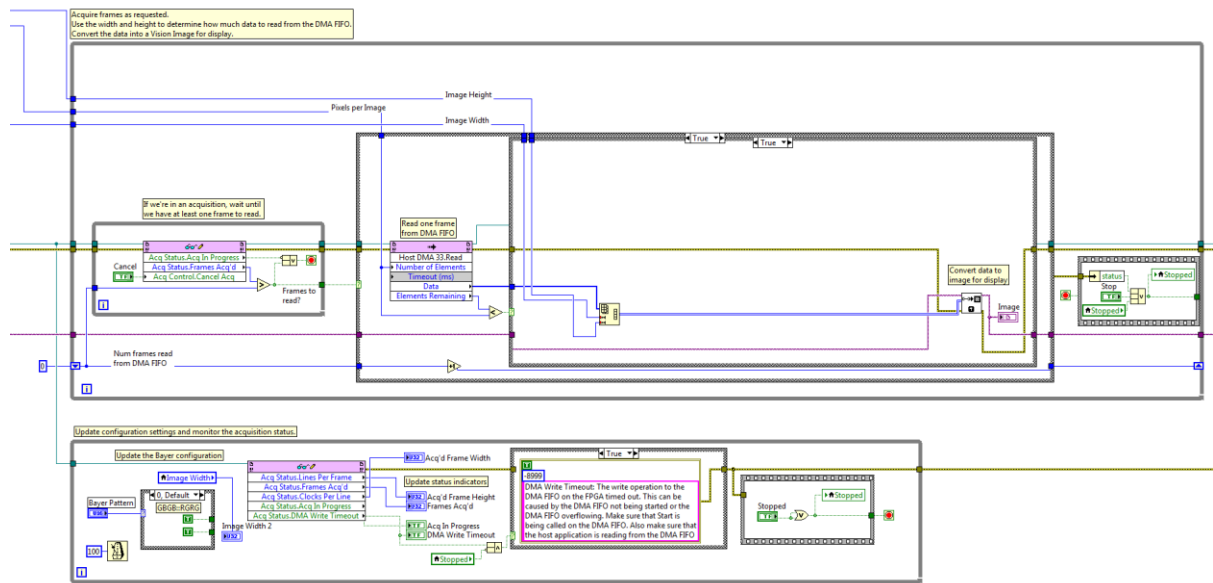
<그림 9. Host 와 FPGA vi 실행화면>

- Block diagram : Host.vi (순서대로 연속됨)

: Host vi : 메모리에서 데이터 받아 출력







<그림 10. Host.vi 화면>

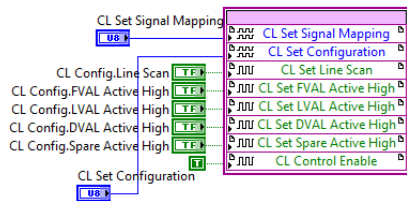
- Block diagram : FPGA.vi (순서대로 연속됨)

: 카메라로부터 영상을 입력받고 Tap 수를 지정한 후 직접 메모리에 접근

### Configure Camera Link

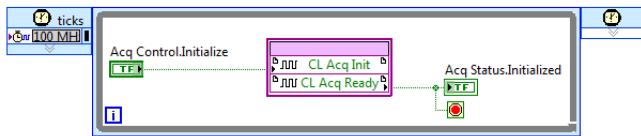
Here we configure the Camera Link settings. This includes the configuration type (base, medium, or full/extended), the signal mapping (standard or a 10-tap configuration), and whether or not the camera is line scan. We also specify the polarity of the FVAL, LVAL, DVAL, and Spare Camera Link signals from the camera that will be used. These configuration options should be set prior to invoking the FPGA "Run" method in the host VI.

In this example, the Camera Link control lines are also enabled here.



### Initialize Aquisition

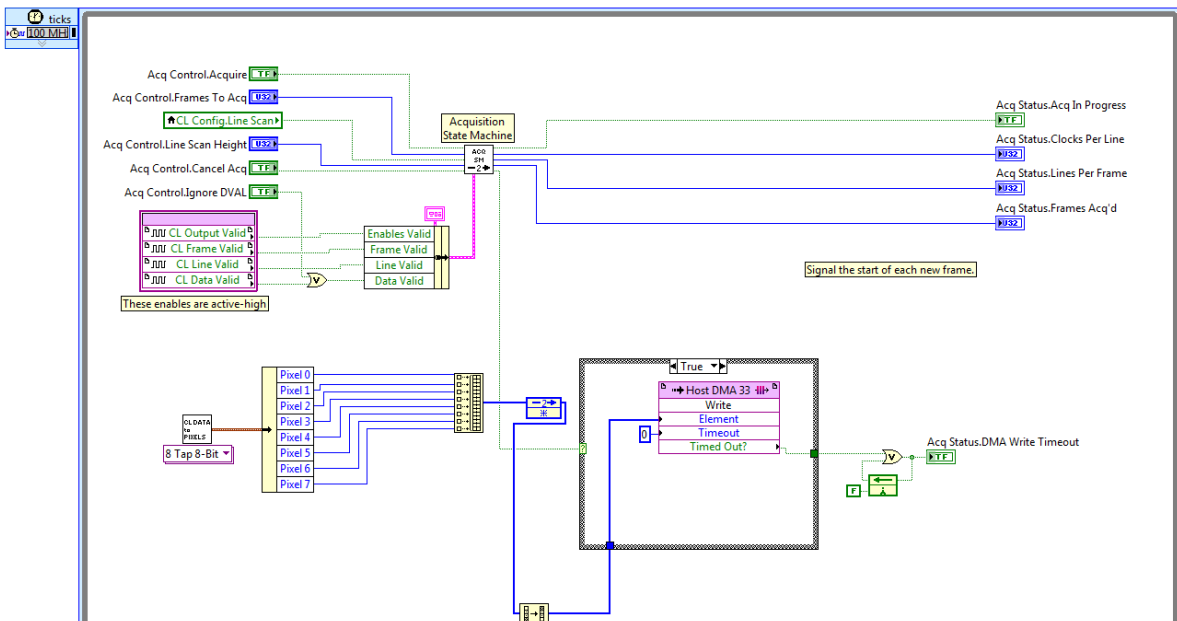
Here we are preparing the Camera Link interface for an acquisition. Setting CL Acq Init to TRUE will begin the initialization sequence, which will try to detect the pixel clock from the camera. If the pixel clock is detected successfully, then CL Acq Ready will become TRUE and end the initialization sequence.



### Acquisition

This is the acquisition loop that is responsible for acquiring and streaming image data back to the host as well as updating the acquisition status indicators. This example supports continuous and finite acquisitions for both area scan and line scan cameras. Camera data is written to the DMA FIFO only when the camera data is part of the image currently being acquired. Otherwise, the data is ignored.

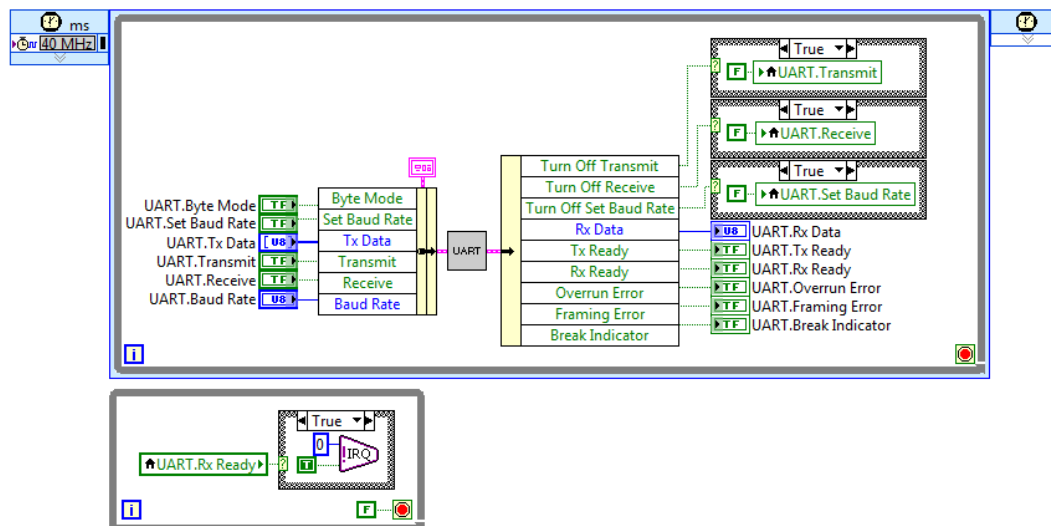
The clock source for this timed loop needs to be the same as the clock selected for the "Image Data Clock" of the IO Module. The "Image Data Clock" needs to be faster than the Pixel Clock of the camera in order to acquire all image data.



## Serial Interface

Drop these loops into any VI to access the Camera Link serial interface. Note that changing the functionality of Serial Interface.vi or any of the UART control names could prevent Serial Server.vi on the host side from working properly.

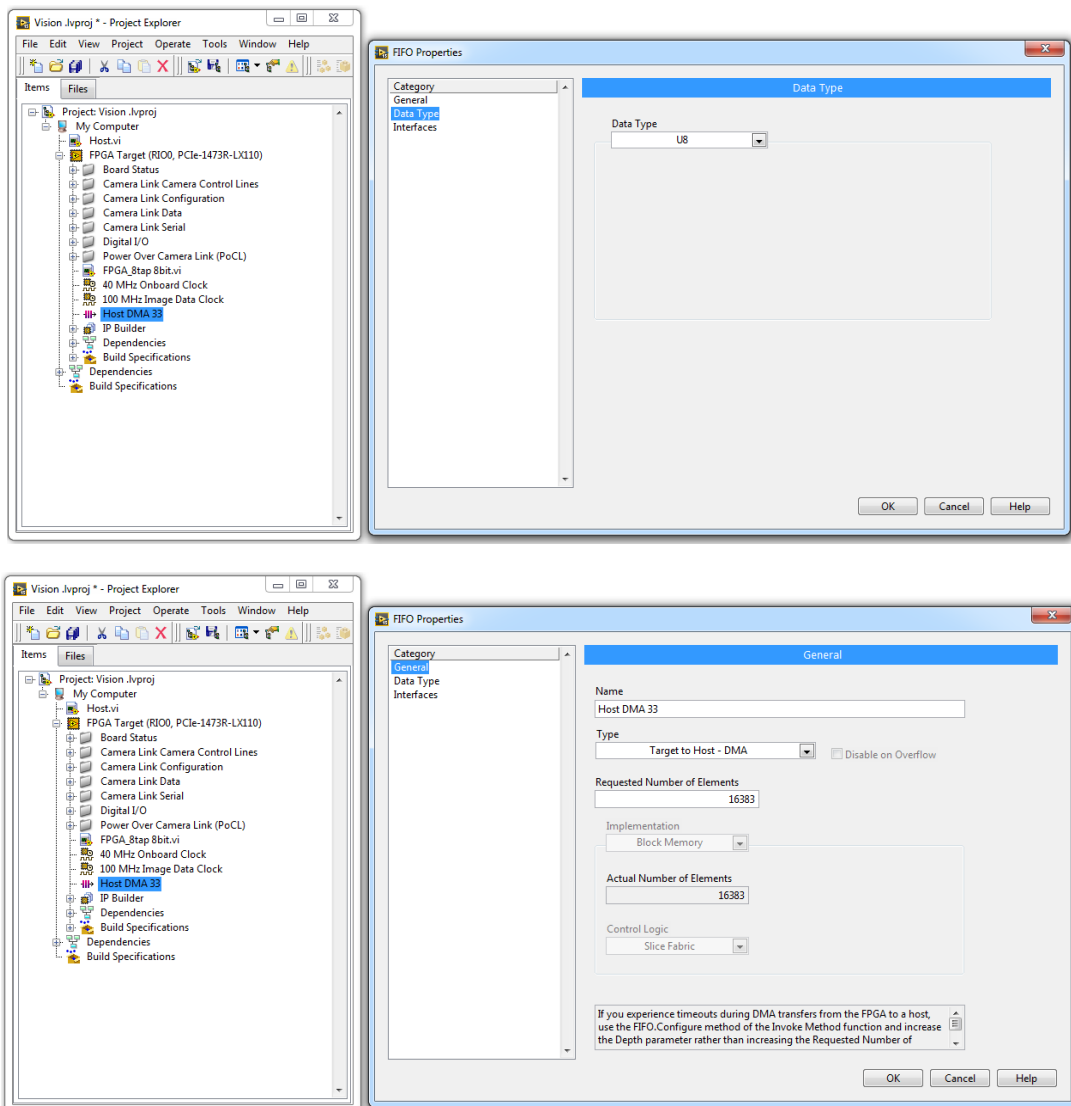
The clock source for this timed loop must be the 40 MHz clock, which is the clock used for "IO Clock - 40 MHz" of the IO Module.

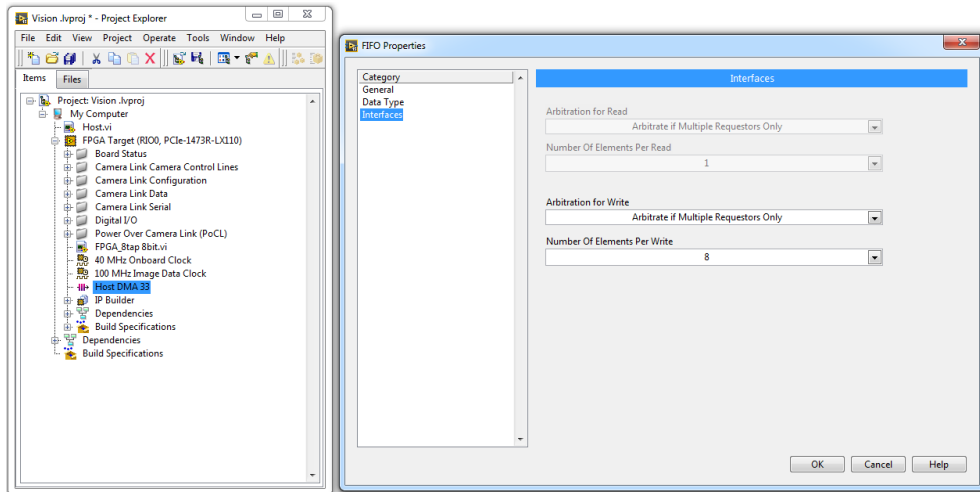


<그림 11. FPGA vi 화면>

## B. Labview project 설정

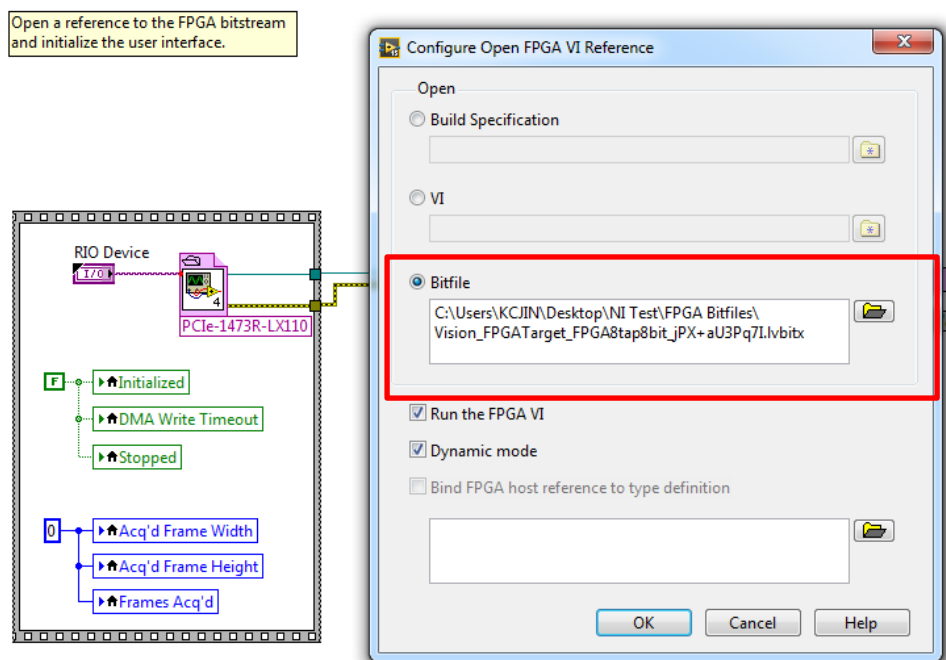
### - Host DMA 33 의 Properties 설정





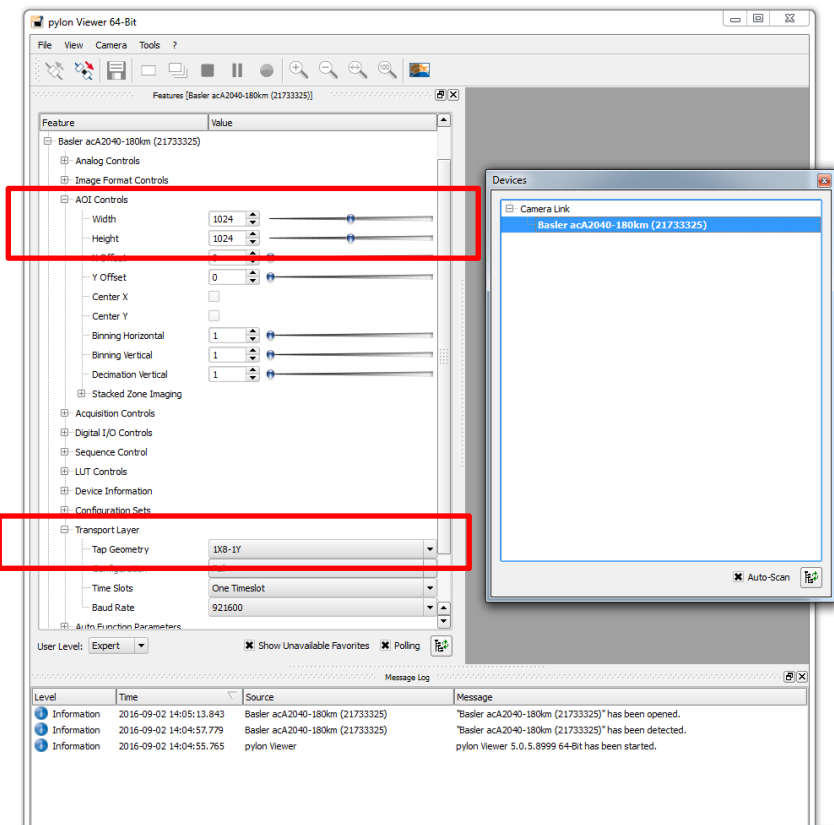
<그림 12. Host DMA 설정 화면>

- Bit 파일은 FPGA vi 컴파일시 생성되며, 생성된 Bit 파일은 <그림 13>와 같이 Host.vi 코드에서 경로를 설정해 주어야 함



<그림 13. Bit 파일 경로 설정 화면>

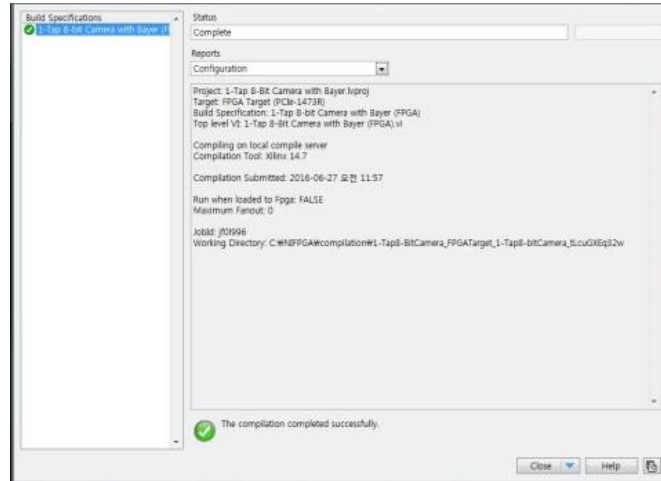
- 카메라 프로그램 설정



<그림 14. 카메라 설정 화면>

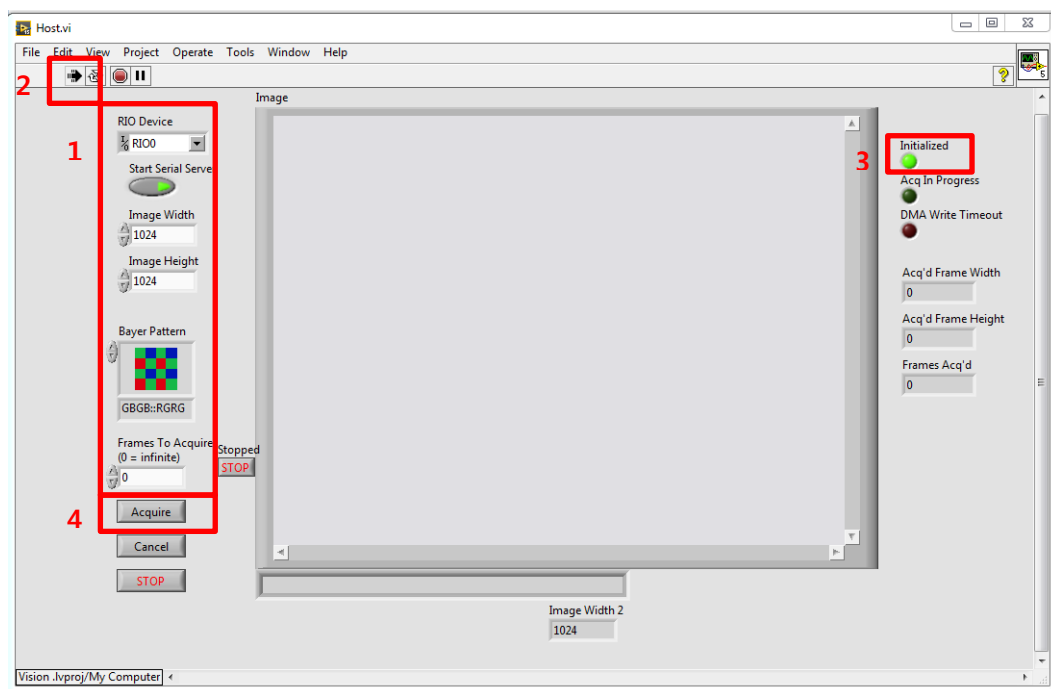
### C. Labview project 실행

- FPGA.vi 프로그램을 실행하면 다음과 같이 컴파일 화면으로 넘어가며 20-40분 정도의 시간이 걸림. 컴파일이 완료 되면 .lvbitx(bit 파일) 파일이 생성됨



<그림 15. FPGA vi 실행 후 컴파일 화면>

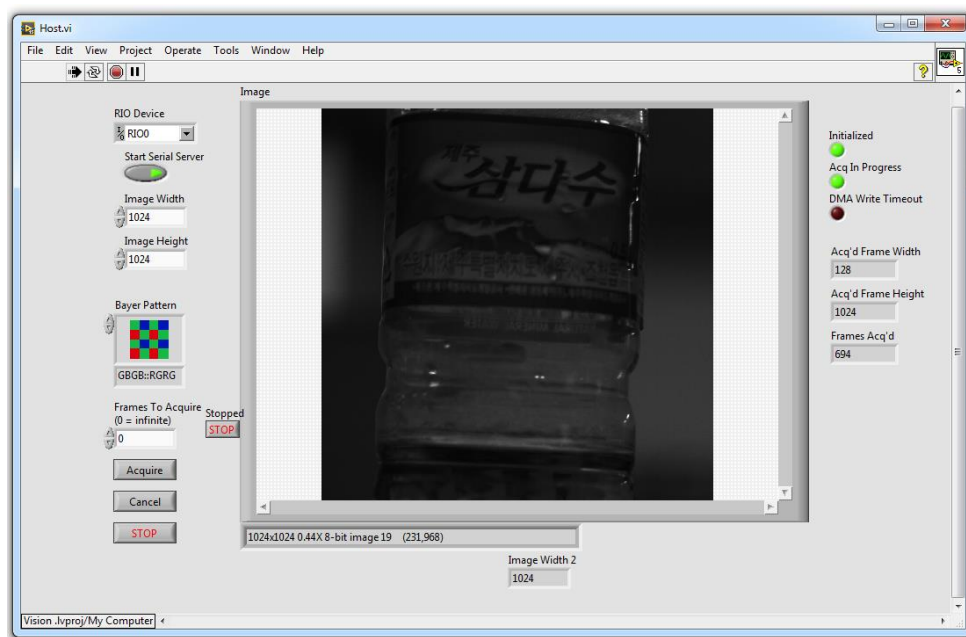
- Host.vi 파일을 실행 후 카메라와의 시리얼 통신을 확인한 후 카메라 프로그램을 통해 이미지의 Width, Height를 변경한 후 acquire 버튼으로 이미지 획득



<그림 16. Host.vi 실행 순서 화면>

#### D. Labview project 수행 결과

- 이미지 크기 : 1024x1024x10bit (1.25MB)
- Frame rate : 298 f/s (298Hz), Host PC 출력 기준
- 8-tap 사용 : 카메라로부터 나오는 데이터 채널을 의미하며 8-tap 은 클럭 한번에 8 픽셀이 출력됨을 의미함
- Bit 파일 생성 : FPGA vi 컴파일 후 생성된 .lvbitx 파일
- 이미지 획득



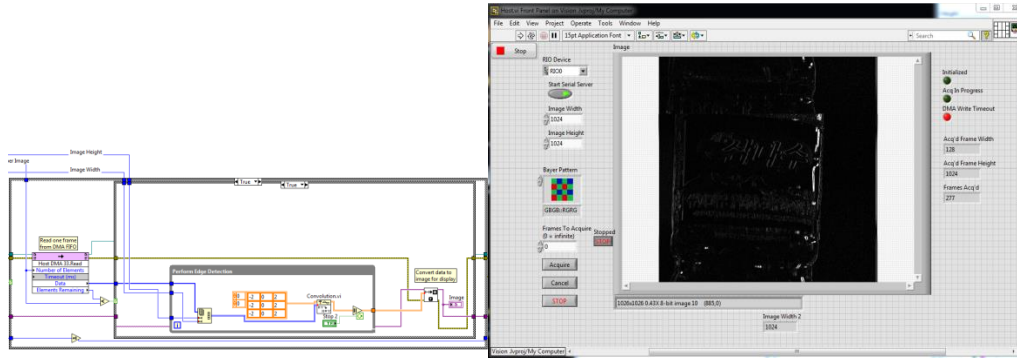
<그림 17. 영상 획득 화면>



#### 4. 영상처리

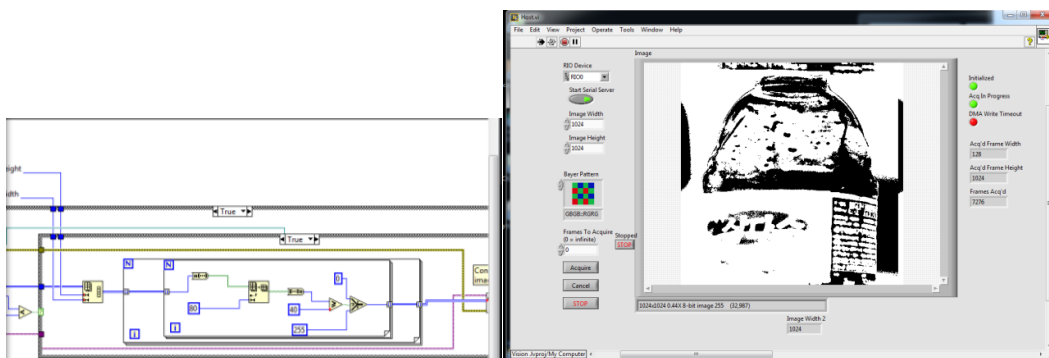
##### A. Host vi 수정 : Edge detection

- 영상에 3X3 mask를 Convolution 하여 영상 edge를 출력함



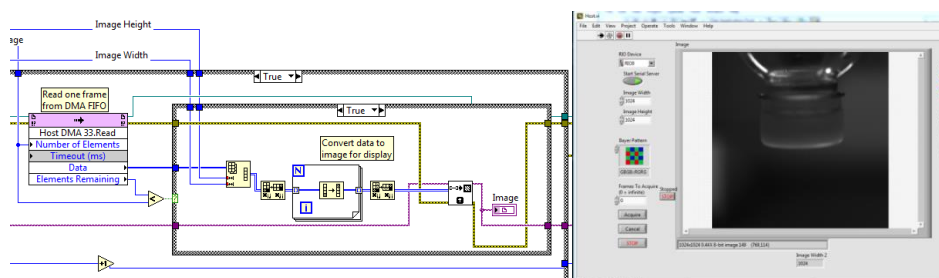
<그림 18. Edge detection 코드와 수정 결과>

##### B. Host vi 수정 : Thresholding



<그림 19. Thresholding 코드와 수정 결과>

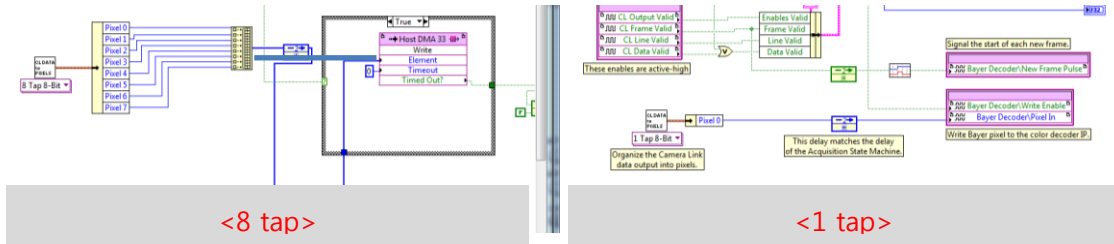
##### C. Host vi 수정 : Reverse



<그림 20. Reverse 코드와 수정 결과>

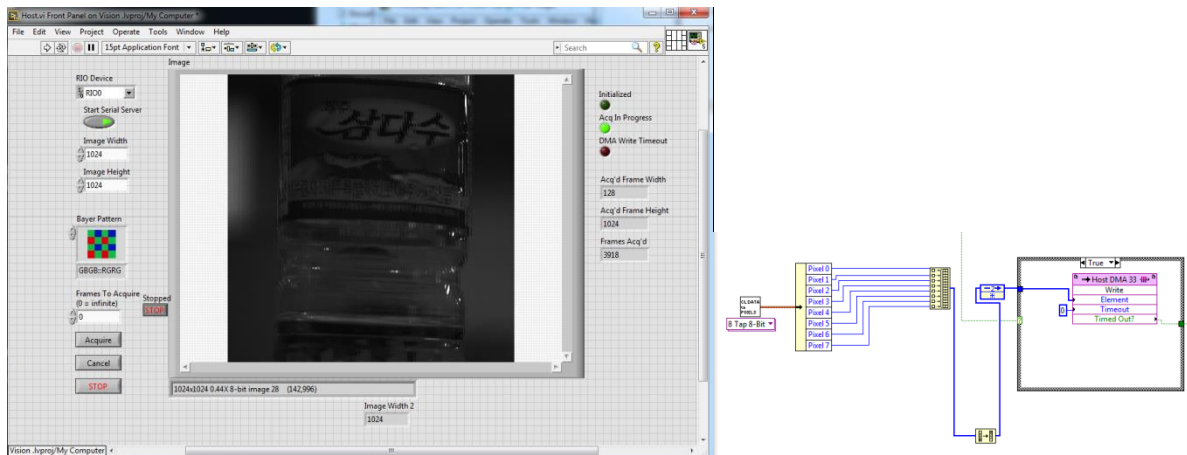
#### D. FPGA vi 수정 : 8tap -1tap

- 현재 시스템에선 8-tap 사용하고 있음



<그림 21. 8Tap과 1Tap의 코드 비교>

#### E. FPGA vi 수정 : reverse

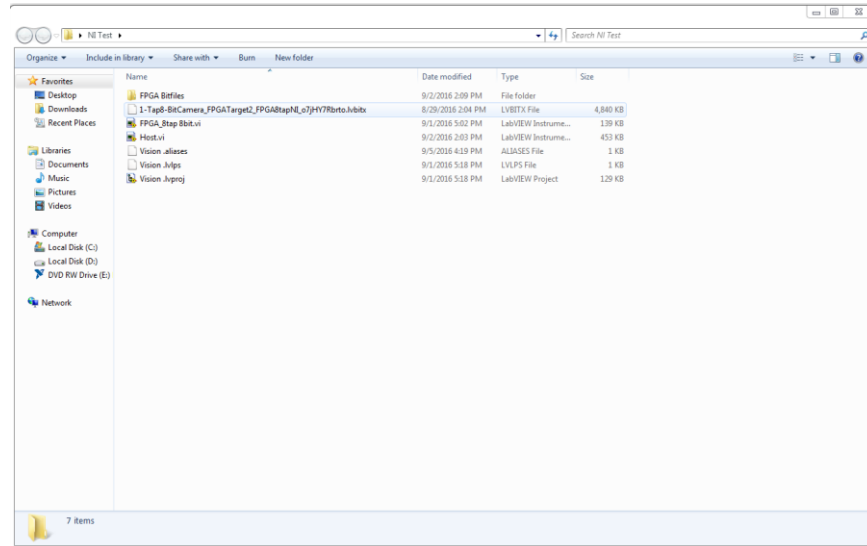


<그림 22. FPGA reverse 코드 화면>

## 5. 부록

### A. 예제 폴더 디렉토리

- c://Desktop/NI Test



<그림 23. 예제프로그램 폴더 화면 >

### B. 버전관리

- Vision\_1\_00.vi ,파일명\_버전\_수정횟수.vi
- 간단한 코드수정 : Vision\_1\_01.vi
- 복잡한 코드수정 : Vision\_1\_10.vi