**GoldenBell2 보드에서 Petalinux 설정**

**Revision 1.0**

**2023. 03. 21**

[**CRZ Technology**](http://www.crz-tech.com/)

**Document History**

|  |  |  |
| --- | --- | --- |
| **Revision** | **Date** | **Change note** |
| **1.0** | **2023.03.21** | **First Draft** |
|  |  |  |
|  |  |  |
|  |  |  |

**목 차**

[1. 개요 4](#_Toc130303641)

[2. Petalinux 설치 4](#_Toc130303642)

[3. Petalinux 프로젝트 생성 및 빌드 4](#_Toc130303643)

[4. SD Card Fusing 방법 5](#_Toc130303644)

[5. U-Boot와 Kernel 이미지 교체 방법 5](#_Toc130303645)

[6. Ethernet 검증 5](#_Toc130303646)

[7. QSPI 검증 7](#_Toc130303647)

[8. USB device 검증 8](#_Toc130303648)

[9. FND / LED 검증 9](#_Toc130303649)

[10. Key 입력 검증 10](#_Toc130303650)

[11. PCIe 검증 11](#_Toc130303651)

[**11.1.** XDAM 테스트 13](#_Toc130303652)

# **개요**

이 문서는 Ubuntu 18.04.6 LTS 상에서 Xilinx사 ZYNQ-7000 XC7Z015CLG485-2 기반 GoldenBell2 보드에서 Petalinux를 동작시키는 방법을 정리하였다.

# **Petalinux 설치**

아래 링크에서 Petalinux installer 2022.2 버전을 다운받는다.

|  |
| --- |
| <https://www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/embedded-design-tools.html> |

Petalinux Tool을 인스톨한다.

|  |
| --- |
| $ sudo –u <user\_name> mkdir –p /opt/pkg/petalinux-v2022.2  $ ./petalinux-v2022.2-final-installer.run /opt/pkg/petalinux-v2022.2 |

Petalinux Tool을 이용하기 위해서는 아래 명령으로 환경 설정을 한다.

|  |
| --- |
| $ source /opt/pkg/petalinux-v2022.2/settings.sh |

# **Petalinux 프로젝트 생성 및 빌드**

GoldenBell2 보드용 BSP를 복사한다.

|  |
| --- |
| [goldenbell2\_20222\_20230321.bsp](https://drive.google.com/file/d/1d9uSFX5Rr-oj4DwqNZl-H8077_OeTND9/view?usp=sharing) |

GoldenBell2 BSP를 기반으로 하여 프로젝트를 생성한다.

|  |
| --- |
| $ petalinux-create –t project –s goldenbell2\_20222\_20230321.bsp –n goldenbell2 |

생성된 프로젝트 폴더로 이동한다.

|  |
| --- |
| $ cd goldenbell2 |

Hw description file의 경로를 설정한다.

|  |
| --- |
| $ petalinux-config --get-hw-description=./hardware/GoldenBell2\_22.2\_pcie\_20230321/workspace |

SD 부트용 이미지를 빌드한다.

|  |
| --- |
| $ cd project-spec/build  $ ./build.sh |

빌드 결과물은 아래 디렉토리에 저장된다.

|  |
| --- |
| $ project-spec/image/image |

# **SD Card Fusing 방법**

Host에 SD card를 연결하고 project-spec/image 폴더에서 sudo ./create-sdcard.sh를 실행하여 부팅 SD card를 만든다.

|  |
| --- |
| $ cd project-spec/images  $ sudo ./create-sdcard.sh |

GoldenBell2 보드에서 SD Card 부팅을 위해서 SW1를 “101100”으로 설정한다. “ON” 방향이 “0”이다.

Fusing한 SD Card를 GoldenBell2 보드의 SD Card 슬롯에 삽입 후 전원을 인가한다. 시리얼 터미널에 디버그 로그가 나오는지 확인한다. USB Type B to Type A 케이블을 J9에 삽입 후 호스트와 연결한다.

user id로 petalinux를 입력하고 새로운 password를 설정한다.

|  |
| --- |
| PetaLinux 2022.2\_release\_S10071807 GoldenBell2 /dev/ttyPS0  GoldenBell2 login: petalinux  Password:  petalinux@GoldenBell2:~$ |

# U-Boot와 Kernel 이미지 교체 방법

NFS 설정이 되어 있는 경우에 런타임에 u-boot와 kernel 이미지를 디바이스의 콘솔에서 아래와 같이 교체할 수 있다.

|  |
| --- |
| $ sudo ./tmp\_kernel\_update.sh |

# Ethernet 검증

GoldenBell2 보드 부팅 후 petalinux 계정으로 로그인한다. 아래 명령을 수행하여 ethernet이 정상적으로 동작하는지 확인한다. iperf3을 실행하여 전송 성능을 확인한다.

|  |
| --- |
| petalinux@GoldenBell2:~$ ping 192.168.1.2  PING 192.168.1.2 (192.168.1.2): 56 data bytes  64 bytes from 192.168.1.2: seq=0 ttl=64 time=0.656 ms  64 bytes from 192.168.1.2: seq=1 ttl=64 time=0.431 ms  64 bytes from 192.168.1.2: seq=2 ttl=64 time=0.441 ms  64 bytes from 192.168.1.2: seq=3 ttl=64 time=0.411 ms |

|  |
| --- |
| petalinux@GoldenBell2:~$ iperf3 -s  -----------------------------------------------------------  Server listening on 5201  -----------------------------------------------------------  Accepted connection from 192.168.1.8, port 60840  [ 5] local 192.168.1.7 port 5201 connected to 192.168.1.8 port 60842  [ ID] Interval Transfer Bitrate  [ 5] 0.00-1.00 sec 90.0 MBytes 755 Mbits/sec  [ 5] 1.00-2.00 sec 90.2 MBytes 757 Mbits/sec  [ 5] 2.00-3.00 sec 90.5 MBytes 759 Mbits/sec  [ 5] 3.00-4.00 sec 90.4 MBytes 758 Mbits/sec  [ 5] 4.00-5.00 sec 90.4 MBytes 758 Mbits/sec  [ 5] 5.00-6.00 sec 90.1 MBytes 756 Mbits/sec  [ 5] 6.00-7.00 sec 90.1 MBytes 756 Mbits/sec  [ 5] 7.00-8.00 sec 90.2 MBytes 756 Mbits/sec  [ 5] 8.00-9.00 sec 90.3 MBytes 757 Mbits/sec  [ 5] 9.00-10.00 sec 90.3 MBytes 757 Mbits/sec  [ 5] 10.00-10.01 sec 488 KBytes 744 Mbits/sec  - - - - - - - - - - - - - - - - - - - - - - - - -  [ ID] Interval Transfer Bitrate  [ 5] 0.00-10.01 sec 903 MBytes 757 Mbits/sec receiver  -----------------------------------------------------------  Server listening on 5201  -----------------------------------------------------------  ^Ciperf3: interrupt - the server has terminated  petalinux@GoldenBell2:~$ iperf3 -c 192.168.1.8  Connecting to host 192.168.1.8, port 5201  [ 5] local 192.168.1.7 port 41892 connected to 192.168.1.8 port 5201  [ ID] Interval Transfer Bitrate Retr Cwnd  [ 5] 0.00-1.00 sec 67.5 MBytes 565 Mbits/sec 1 174 KBytes  [ 5] 1.00-2.00 sec 67.5 MBytes 566 Mbits/sec 0 182 KBytes  [ 5] 2.00-3.00 sec 67.5 MBytes 567 Mbits/sec 0 182 KBytes  [ 5] 3.00-4.02 sec 68.8 MBytes 567 Mbits/sec 0 185 KBytes  [ 5] 4.02-5.02 sec 67.5 MBytes 566 Mbits/sec 0 189 KBytes  [ 5] 5.02-6.00 sec 66.2 MBytes 565 Mbits/sec 1 182 KBytes  [ 5] 6.00-7.00 sec 67.5 MBytes 565 Mbits/sec 1 188 KBytes  [ 5] 7.00-8.00 sec 67.5 MBytes 566 Mbits/sec 0 188 KBytes  [ 5] 8.00-9.01 sec 67.5 MBytes 565 Mbits/sec 1 177 KBytes  [ 5] 9.01-10.01 sec 67.5 MBytes 566 Mbits/sec 0 181 KBytes  - - - - - - - - - - - - - - - - - - - - - - - - -  [ ID] Interval Transfer Bitrate Retr  [ 5] 0.00-10.01 sec 675 MBytes 566 Mbits/sec 4 sender  [ 5] 0.00-10.01 sec 675 MBytes 566 Mbits/sec receiver  iperf Done.  petalinux@GoldenBell2:~$ |

# QSPI 검증

아래 명령어를 입력하여 QSPI가 정상적으로 인식되어 UBIFS로 마운트되는지 확인한다.

|  |
| --- |
| petalinux@GoldenBell2:~$ sudo ./make\_qspi\_ubifs.sh  Password:  Start qspi ubifs filesystem ...  [ 708.565252] UBIFS (ubi0:0): un-mount UBI device 0  [ 708.570018] UBIFS (ubi0:0): background thread "ubifs\_bgt0\_0" stops  [ 709.090852] ubi0: detaching mtd0  [ 709.096538] ubi0: mtd0 is detached  ubiformat: mtd0 (nor), size 16777216 bytes (16.0 MiB), 256 eraseblocks of 65536 bytes (64.0 KiB), min. I/O size 1 bytes  libscan: scanning eraseblock 255 -- 100 % complete  ubiformat: 256 eraseblocks have valid erase counter, mean value is 2  ubiformat: formatting eraseblock 255 -- 100 % complete  [ 808.980747] ubi0: attaching mtd0  [ 809.049890] ubi0: scanning is finished  [ 809.060854] ubi0: attached mtd0 (name "datafs", size 16 MiB)  [ 809.066867] ubi0: PEB size: 65536 bytes (64 KiB), LEB size: 65408 bytes  [ 809.073491] ubi0: min./max. I/O unit sizes: 1/256, sub-page size 1  [ 809.079729] ubi0: VID header offset: 64 (aligned 64), data offset: 128  [ 809.086277] ubi0: good PEBs: 256, bad PEBs: 0, corrupted PEBs: 0  [ 809.092306] ubi0: user volume: 0, internal volumes: 1, max. volumes count: 128  [ 809.099551] ubi0: max/mean erase counter: 5/3, WL threshold: 4096, image sequence number: 781184611  [ 809.108621] ubi0: available PEBs: 252, total reserved PEBs: 4, PEBs reserved for bad PEB handling: 0  [ 809.117793] ubi0: background thread "ubi\_bgt0d" started, PID 495  UBI device number 0, total 256 LEBs (16744448 bytes, 15.9 MiB), available 252 LEBs (16482816 bytes, 15.7 MiB), LEB size 65408 bytes (63.8 KiB)  [Step 1] mtd0 ubiattach done ...  Set volume size to 16482816  Volume ID 0, size 252 LEBs (16482816 bytes, 15.7 MiB), LEB size 65408 bytes (63.8 KiB), dynamic, name "datafs", alignment 1  [Step 2] mtd0 ubimkvol done ...  [ 810.223060] UBIFS (ubi0:0): default file-system created  [ 810.228709] UBIFS (ubi0:0): Mounting in unauthenticated mode  [ 810.234788] UBIFS (ubi0:0): background thread "ubifs\_bgt0\_0" started, PID 498  [ 810.263809] UBIFS (ubi0:0): UBIFS: mounted UBI device 0, volume 0, name "datafs"  [ 810.271218] UBIFS (ubi0:0): LEB size: 65408 bytes (63 KiB), min./max. I/O unit sizes: 8 bytes/256 bytes  [ 810.280652] UBIFS (ubi0:0): FS size: 15828736 bytes (15 MiB, 242 LEBs), max 252 LEBs, journal size 784896 bytes (0 MiB, 12 LEBs)  [ 810.292294] UBIFS (ubi0:0): reserved for root: 747629 bytes (730 KiB)  [ 810.298810] UBIFS (ubi0:0): media format: w5/r0 (latest is w5/r0), UUID 9F5548D4-55AC-4865-9DD7-DEEACB458CF4, small LPT model  [Step 3] mount done ...  Complete ubifs filesystem ...  sync  petalinux@GoldenBell2:~$ df  Filesystem 1K-blocks Used Available Use% Mounted on  /dev/root 6642224 27212 6273508 0% /  devtmpfs 505436 4 505432 0% /dev  tmpfs 514140 224 513916 0% /run  tmpfs 514140 56 514084 0% /var/volatile  /dev/mmcblk0p1 1046512 9244 1037268 1% /run/media/mmcblk0p1  ubi0:datafs 13532 20 12784 0% /mnt/qspi\_datafs  petalinux@GoldenBell2:~$ |

# USB device 검증

USB Type B to Type A 케이블을 J8에 삽입하고 호스트와 연결한다.

아래 명령어를 입력하여 USB Mass Storage가 Host에서 정상적으로 인식되는지 확인한다.

|  |
| --- |
| petalinux@GoldenBell2:~$ sudo ./ums\_mount.sh  Password:  10+0 records in  10+0 records out  [ 197.897392] Mass Storage Function, version: 2009/09/11  [ 197.902540] LUN: removable file: (no medium)  petalinux@GoldenBell2:~$ |

# FND / LED 검증

아래 명령어를 입력하여 FND와 LED가 정상적으로 켜지는지 확인한다.

|  |
| --- |
| petalinux@GoldenBell2:~$ sudo ./fnd\_led\_test.sh  gpio 999 on  gpio 1000 on  gpio 1001 on  gpio 1002 on  gpio 1003 on  gpio 1004 on  gpio 1005 on  gpio 1006 on  gpio 1007 on  gpio 1008 on  gpio 1009 on  gpio 1010 on  gpio 1011 on  gpio 1012 on  gpio 1013 on  gpio 986 on  gpio 1019 on  gpio 1020 on  gpio 1021 on  gpio 1022 on  gpio 1023 on  fnd 995 on  fnd 996 on  fnd 997 on  fnd 998 on  gpio 999 off  gpio 1000 off  gpio 1001 off  gpio 1002 off  gpio 1003 off  gpio 1004 off  gpio 1005 off  gpio 1006 off  gpio 1007 off  gpio 1008 off  gpio 1009 off  gpio 1010 off  gpio 1011 off  gpio 1012 off  gpio 1013 off  gpio 986 off  gpio 1019 off  gpio 1020 off  gpio 1021 off  gpio 1022 off  gpio 1023 off  petalinux@GoldenBell2:~$ |

# Key 입력 검증

아래 명령어를 입력하고 SW2 ~ SW6을 눌러 인식되는지 확인한다.

|  |
| --- |
| petalinux@GoldenBell2:~$ sudo evtest  No device specified, trying to scan all of /dev/input/event\*  Available devices:  /dev/input/event0: gpio-keys-polled  Select the device event number [0-0]: 0  Input driver version is 1.0.1  Input device ID: bus 0x19 vendor 0x1 product 0x1 version 0x100  Input device name: "gpio-keys-polled"  Supported events:  Event type 0 (EV\_SYN)  Event type 1 (EV\_KEY)  Event code 3 (KEY\_2)  Event code 4 (KEY\_3)  Event code 5 (KEY\_4)  Event code 6 (KEY\_5)  Event code 7 (KEY\_6)  Key repeat handling:  Repeat type 20 (EV\_REP)  Repeat code 0 (REP\_DELAY)  Value 250  Repeat code 1 (REP\_PERIOD)  Value 33  Properties:  Testing ... (interrupt to exit)  Event: time 1520615455.492777, type 1 (EV\_KEY), code 3 (KEY\_2), value 1  Event: time 1520615455.492777, -------------- SYN\_REPORT ------------  Event: time 1520615455.712755, type 1 (EV\_KEY), code 3 (KEY\_2), value 0  Event: time 1520615455.712755, -------------- SYN\_REPORT ------------  Event: time 1520615456.372770, type 1 (EV\_KEY), code 4 (KEY\_3), value 1  Event: time 1520615456.372770, -------------- SYN\_REPORT ------------  Event: time 1520615456.592768, type 1 (EV\_KEY), code 4 (KEY\_3), value 0  Event: time 1520615456.592768, -------------- SYN\_REPORT ------------  Event: time 1520615457.142774, type 1 (EV\_KEY), code 5 (KEY\_4), value 1  Event: time 1520615457.142774, -------------- SYN\_REPORT ------------  Event: time 1520615457.362759, type 1 (EV\_KEY), code 5 (KEY\_4), value 0  Event: time 1520615457.362759, -------------- SYN\_REPORT ------------  Event: time 1520615457.802777, type 1 (EV\_KEY), code 6 (KEY\_5), value 1  Event: time 1520615457.802777, -------------- SYN\_REPORT ------------  Event: time 1520615458.022766, type 1 (EV\_KEY), code 6 (KEY\_5), value 0  Event: time 1520615458.022766, -------------- SYN\_REPORT ------------  Event: time 1520615458.682778, type 1 (EV\_KEY), code 7 (KEY\_6), value 1  Event: time 1520615458.682778, -------------- SYN\_REPORT ------------  Event: time 1520615458.902768, type 1 (EV\_KEY), code 7 (KEY\_6), value 0  Event: time 1520615458.902768, -------------- SYN\_REPORT ------------  ^Cpetalinux@GoldenBell2:~$ |

# PCIe 검증

GoldenBell2 보드를 PCIe 슬롯에 장착한다. 호스트 PC 전원을 인가하여 PCIe 링크가 정상적인지확인한다. 링크가 정상적이면 LD17 LED가 켜진다.

호스트 리눅스에 로그인한 후 lspci를 실행하여 링크가 설정이 되었는지 확인한다.

|  |
| --- |
| hgchoe@ubuntu:~$ sudo lspci -vvv -s 01:00.0  [sudo] password for hgchoe:  01:00.0 Serial controller: Xilinx Corporation Device 7024 (prog-if 01 [16450])  Subsystem: Xilinx Corporation Device 0007  Control: I/O- Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR+ FastB2B- DisINTx-  Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR- INTx-  Latency: 0  Interrupt: pin A routed to IRQ 128  Region 0: Memory at dea00000 (32-bit, non-prefetchable) [size=64K]  Capabilities: [40] Power Management version 3  Flags: PMEClk- DSI- D1- D2- AuxCurrent=0mA PME(D0+,D1+,D2+,D3hot+,D3cold-)  Status: D0 NoSoftRst+ PME-Enable- DSel=0 DScale=0 PME-  Capabilities: [48] MSI: Enable+ Count=1/1 Maskable- 64bit+  Address: 00000000fee002d8 Data: 0000  Capabilities: [60] Express (v2) Endpoint, MSI 00  DevCap: MaxPayload 512 bytes, PhantFunc 0, Latency L0s <64ns, L1 unlimited  ExtTag+ AttnBtn- AttnInd- PwrInd- RBE+ FLReset- SlotPowerLimit 75.000W  DevCtl: Report errors: Correctable- Non-Fatal- Fatal- Unsupported-  RlxdOrd+ ExtTag+ PhantFunc- AuxPwr- NoSnoop+  MaxPayload 256 bytes, MaxReadReq 512 bytes  DevSta: CorrErr- UncorrErr- FatalErr- UnsuppReq- AuxPwr- TransPend-  LnkCap: Port #0, Speed 5GT/s, Width x4, ASPM L0s, Exit Latency L0s unlimited, L1 unlimited  ClockPM- Surprise- LLActRep- BwNot- ASPMOptComp-  LnkCtl: ASPM Disabled; RCB 64 bytes Disabled- CommClk+  ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt-  LnkSta: Speed 5GT/s, Width x4, TrErr- Train- SlotClk+ DLActive- BWMgmt- ABWMgmt-  DevCap2: Completion Timeout: Range B, TimeoutDis-, LTR-, OBFF Not Supported  DevCtl2: Completion Timeout: 50us to 50ms, TimeoutDis-, LTR-, OBFF Disabled  LnkCtl2: Target Link Speed: 5GT/s, EnterCompliance- SpeedDis-  Transmit Margin: Normal Operating Range, EnterModifiedCompliance- ComplianceSOS-  Compliance De-emphasis: -6dB  LnkSta2: Current De-emphasis Level: -6dB, EqualizationComplete-, EqualizationPhase1-  EqualizationPhase2-, EqualizationPhase3-, LinkEqualizationRequest-  Capabilities: [100 v1] Device Serial Number 00-00-00-00-00-00-00-00  Kernel driver in use: xdma  Kernel modules: xdma  hgchoe@ubuntu:~$ |

## XDAM 테스트

아래 링크를 접근하여 필요한 파일을 우분투 18.04 호스트에 복사한다.

|  |
| --- |
| <https://github.com/Xilinx/dma_ip_drivers> |

다음 명령을 실행하여 XDMA 커널 모듈과 응용 프로그램을 빌드한다.

|  |
| --- |
| $ cd XDMA/linux-kernel/  $ cd xdma  $ make install  $ cd ../tools  $ make  $ cd ../tests |

XDMA 커널 모듈을 로드하고 XDMA가 동작하는지 확인한다.

|  |
| --- |
| hgchoe@ubuntu:~/project/XDMA/dma\_ip\_drivers-master/XDMA/linux-kernel/tests$ sudo ./load\_driver.sh 4  [sudo] password for hgchoe:  interrupt\_selection 4.  xdma 86016 0  Loading driver...insmod xdma.ko poll\_mode=1 ...  The Kernel module installed correctly and the xmda devices were recognized.  DONE  hgchoe@ubuntu:~/project/XDMA/dma\_ip\_drivers-master/XDMA/linux-kernel/tests$ sudo ./run\_test.sh  Info: Number of enabled h2c channels = 2  Info: Number of enabled c2h channels = 2  Info: The PCIe DMA core is memory mapped.  Info: Running PCIe DMA memory mapped write read test  transfer size: 4096, count: 16  Info: Writing to h2c channel 0 at address offset 0.  Info: Writing to h2c channel 1 at address offset 4096.  Info: Wait for current transactions to complete.  /dev/xdma0\_h2c\_0 \*\* Average BW = 4096, 159.337128  /dev/xdma0\_h2c\_1 \*\* Average BW = 4096, 93.405090  Info: Writing to h2c channel 0 at address offset 8192.  Info: Writing to h2c channel 1 at address offset 12288.  Info: Wait for current transactions to complete.  /dev/xdma0\_h2c\_0 \*\* Average BW = 4096, 182.120331  /dev/xdma0\_h2c\_1 \*\* Average BW = 4096, 138.185226  Info: Reading from c2h channel 0 at address offset 0.  Info: Reading from c2h channel 1 at address offset 4096.  Info: Wait for current transactions to complete.  /dev/xdma0\_c2h\_0 \*\* Average BW = 4096, 58.835255  /dev/xdma0\_c2h\_1 \*\* Average BW = 4096, 65.488129  Info: Reading from c2h channel 0 at address offset 8192.  Info: Reading from c2h channel 1 at address offset 12288.  Info: Wait for current transactions to complete.  /dev/xdma0\_c2h\_0 \*\* Average BW = 4096, 31.719082  /dev/xdma0\_c2h\_1 \*\* Average BW = 4096, 73.185005  Info: Checking data integrity.  Info: Data check passed for address range 0 - 4096  Info: Data check passed for address range 4096 - 8192  Info: Data check passed for address range 8192 - 12288  Info: Data check passed for address range 12288 - 16384  Info: All PCIe DMA memory mapped tests passed.  Info: All tests in run\_tests.sh passed.  hgchoe@ubuntu:~/project/XDMA/dma\_ip\_drivers-master/XDMA/linux-kernel/tests$ |