SAMSUNG ARTIK™ Modules

0

ARTIK 053 튜토리얼

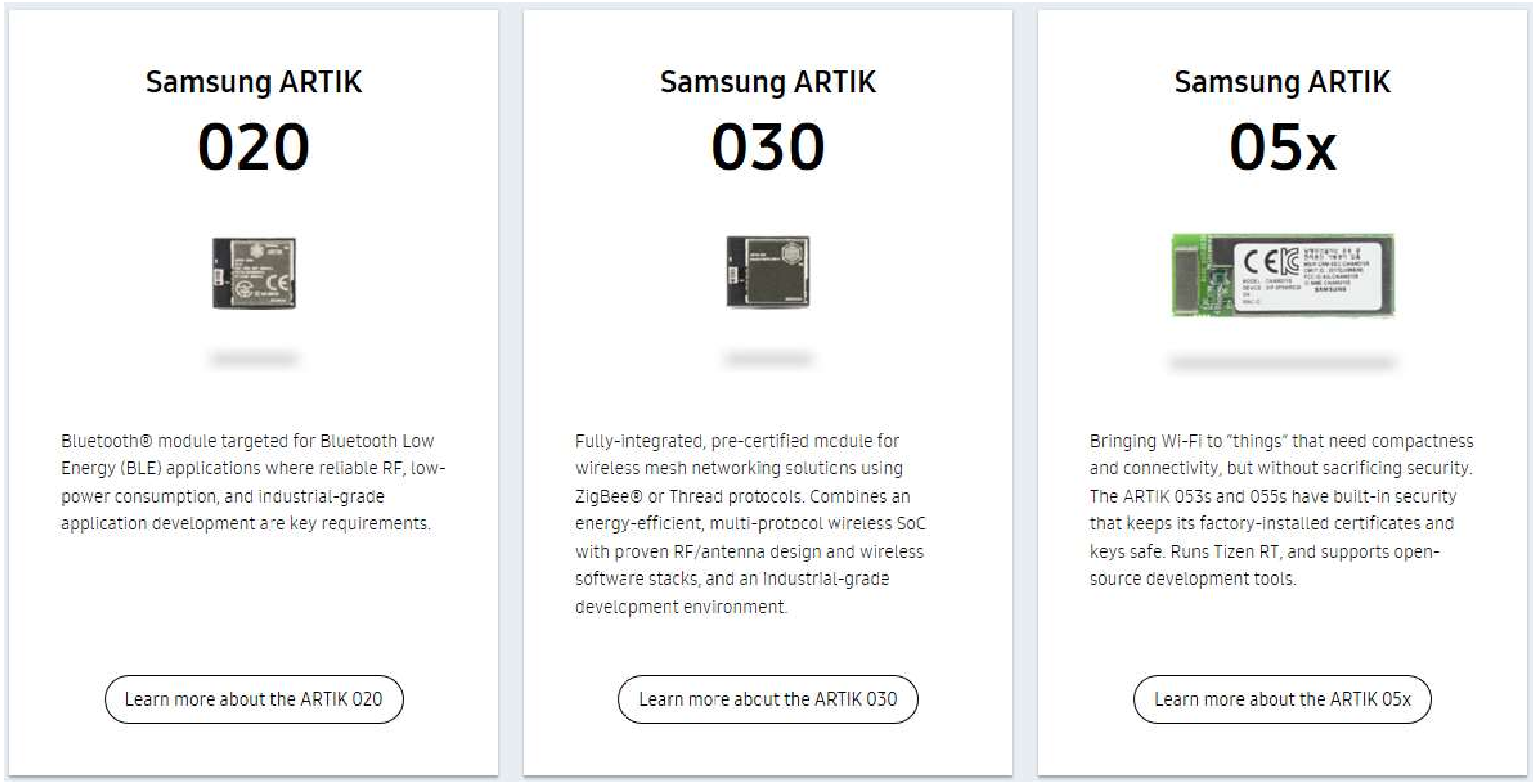
ARTIK 053
Tutorial

Contents

ARTIK 053

ARTIK 053

❖ ARTIK 0 Family



Samsung ARTIK
020

Bluetooth® module targeted for Bluetooth Low Energy (BLE) applications where reliable RF, low-power consumption, and industrial-grade application development are key requirements.

Learn more about the ARTIK 020

Samsung ARTIK
030

Fully-integrated, pre-certified module for wireless mesh networking solutions using ZigBee® or Thread protocols. Combines an energy-efficient, multi-protocol wireless SoC with proven RF/antenna design and wireless software stacks, and an industrial-grade development environment.

Learn more about the ARTIK 030

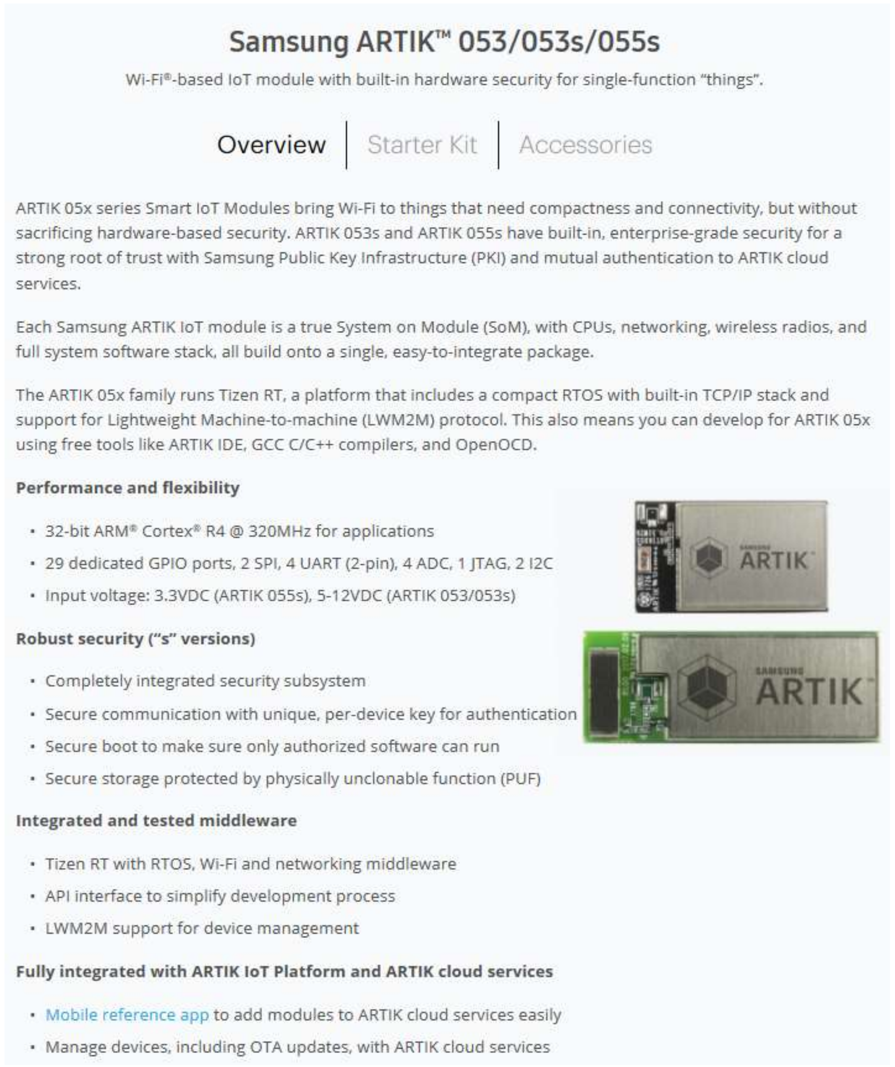
Samsung ARTIK
05x

Bringing Wi-Fi to "things" that need compactness and connectivity, but without sacrificing security. The ARTIK 053s and 055s have built-in security that keeps its factory-installed certificates and keys safe. Runs Tizen RT, and supports open-source development tools.

Learn more about the ARTIK 05x

ARTIK 053

❖ **ARTIK 053 module**



Samsung ARTIK™ 053/053s/055s

Wi-Fi®-based IoT module with built-in hardware security for single-function "things".

Overview | Starter Kit | Accessories

ARTIK 05x series Smart IoT Modules bring Wi-Fi to things that need compactness and connectivity, but without sacrificing hardware-based security. ARTIK 053s and ARTIK 055s have built-in, enterprise-grade security for a strong root of trust with Samsung Public Key Infrastructure (PKI) and mutual authentication to ARTIK cloud services.

Each Samsung ARTIK IoT module is a true System on Module (SoM), with CPUs, networking, wireless radios, and full system software stack, all build onto a single, easy-to-integrate package.

The ARTIK 05x family runs Tizen RT, a platform that includes a compact RTOS with built-in TCP/IP stack and support for Lightweight Machine-to-machine (LWM2M) protocol. This also means you can develop for ARTIK 05x using free tools like ARTIK IDE, GCC C/C++ compilers, and OpenOCD.

Performance and flexibility

- 32-bit ARM® Cortex® R4 @ 320MHz for applications
- 29 dedicated GPIO ports, 2 SPI, 4 UART (2-pin), 4 ADC, 1 JTAG, 2 I2C
- Input voltage: 3.3VDC (ARTIK 055s), 5-12VDC (ARTIK 053/053s)

Robust security ("s" versions)

- Completely integrated security subsystem
- Secure communication with unique, per-device key for authentication
- Secure boot to make sure only authorized software can run
- Secure storage protected by physically unclonable function (PUF)

Integrated and tested middleware

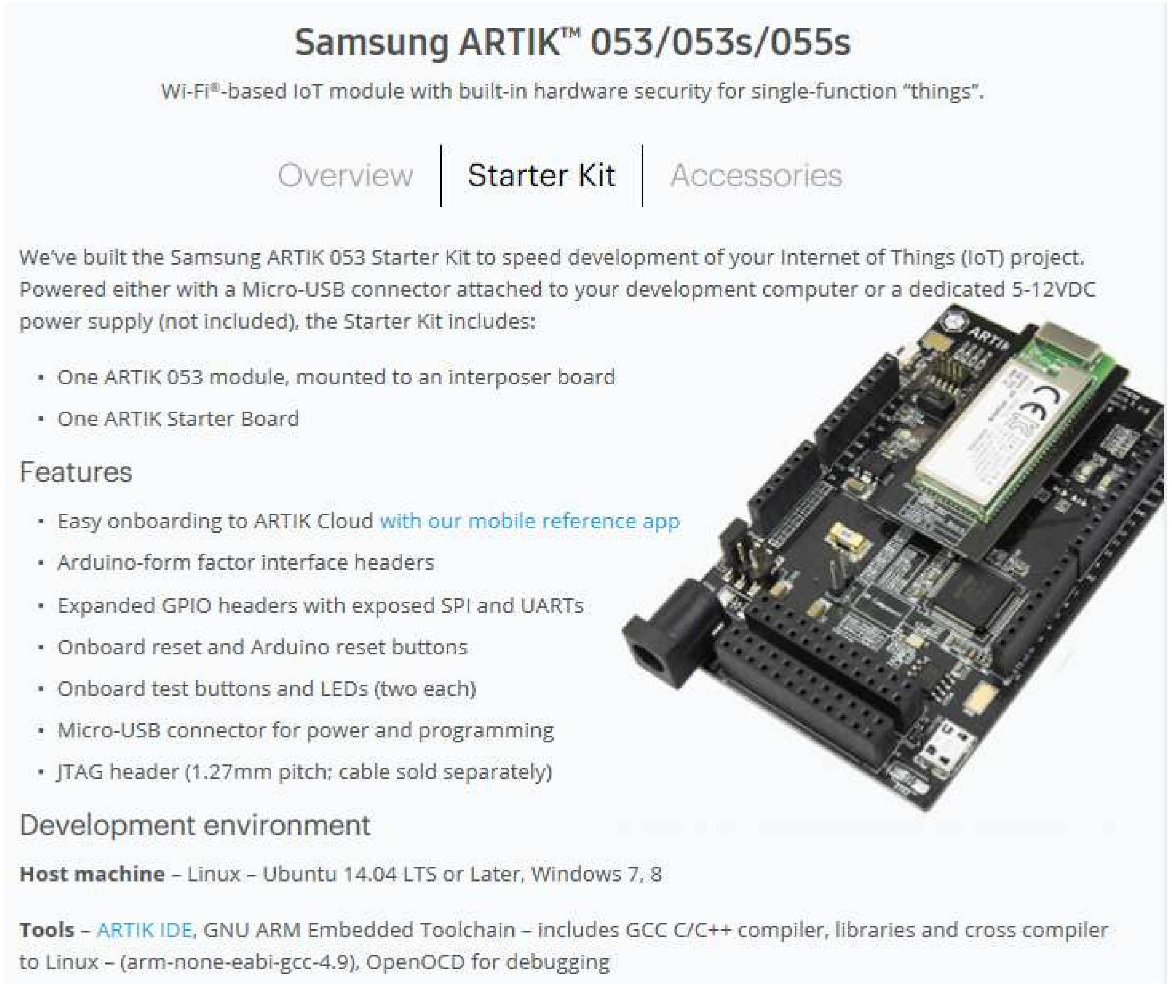
- Tizen RT with RTOS, Wi-Fi and networking middleware
- API interface to simplify development process
- LWM2M support for device management

Fully integrated with ARTIK IoT Platform and ARTIK cloud services

- Mobile reference app to add modules to ARTIK cloud services easily
- Manage devices, including OTA updates, with ARTIK cloud services

ARTIK 053

❖ ARTIK 053 Starter Kit



Samsung ARTIK™ 053/053s/055s

Wi-Fi®-based IoT module with built-in hardware security for single-function "things".

Overview | Starter Kit | Accessories

We've built the Samsung ARTIK 053 Starter Kit to speed development of your Internet of Things (IoT) project. Powered either with a Micro-USB connector attached to your development computer or a dedicated 5-12VDC power supply (not included), the Starter Kit includes:

- One ARTIK 053 module, mounted to an interposer board
- One ARTIK Starter Board

Features

- Easy onboarding to ARTIK Cloud with our mobile reference app
- Arduino-form factor interface headers
- Expanded GPIO headers with exposed SPI and UARTs
- Onboard reset and Arduino reset buttons
- Onboard test buttons and LEDs (two each)
- Micro-USB connector for power and programming
- JTAG header (1.27mm pitch; cable sold separately)

Development environment

Host machine – Linux – Ubuntu 14.04 LTS or Later, Windows 7, 8

Tools – ARTIK IDE, GNU ARM Embedded Toolchain – includes GCC C/C++ compiler, libraries and cross compiler to Linux – (arm-none-eabi-gcc-4.9), OpenOCD for debugging

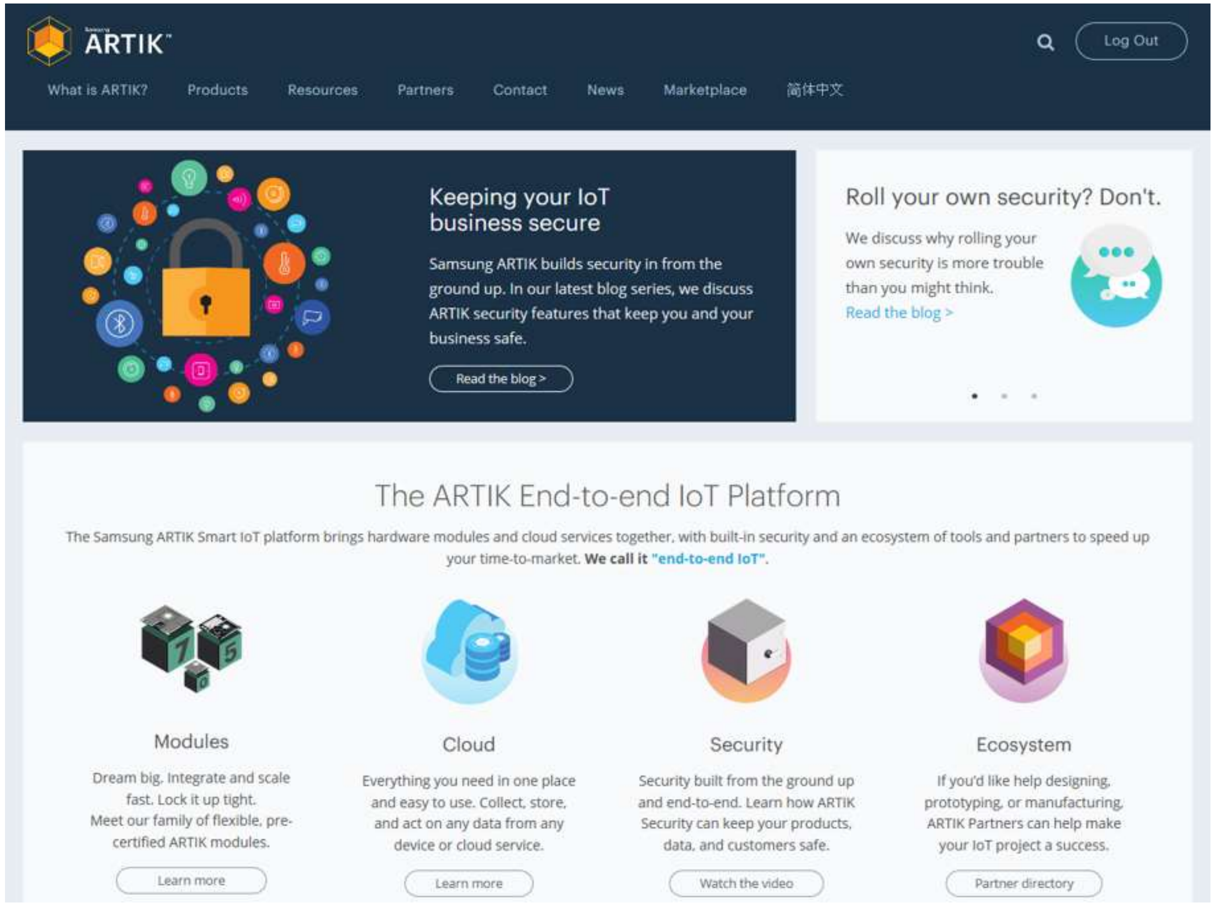
CONTROL IN ROBOTICS

개발환경 구축

– ARTIK IDE –

ARTIK IDE 설치(1)

❖ www.artik.io 접속



What is ARTIK? Products Resources Partners Contact News Marketplace 简体中文

Keeping your IoT business secure

Samsung ARTIK builds security in from the ground up. In our latest blog series, we discuss ARTIK security features that keep you and your business safe.

Read the blog >

Roll your own security? Don't.

We discuss why rolling your own security is more trouble than you might think.

Read the blog >

The ARTIK End-to-end IoT Platform

The Samsung ARTIK Smart IoT platform brings hardware modules and cloud services together, with built-in security and an ecosystem of tools and partners to speed up your time-to-market. **We call it "end-to-end IoT".**

Modules

Dream big. Integrate and scale fast. Lock it up tight. Meet our family of flexible, pre-certified ARTIK modules.

Learn more

Cloud

Everything you need in one place and easy to use. Collect, store, and act on any data from any device or cloud service.

Learn more

Security

Security built from the ground up and end-to-end. Learn how ARTIK Security can keep your products, data, and customers safe.

Watch the video

Ecosystem

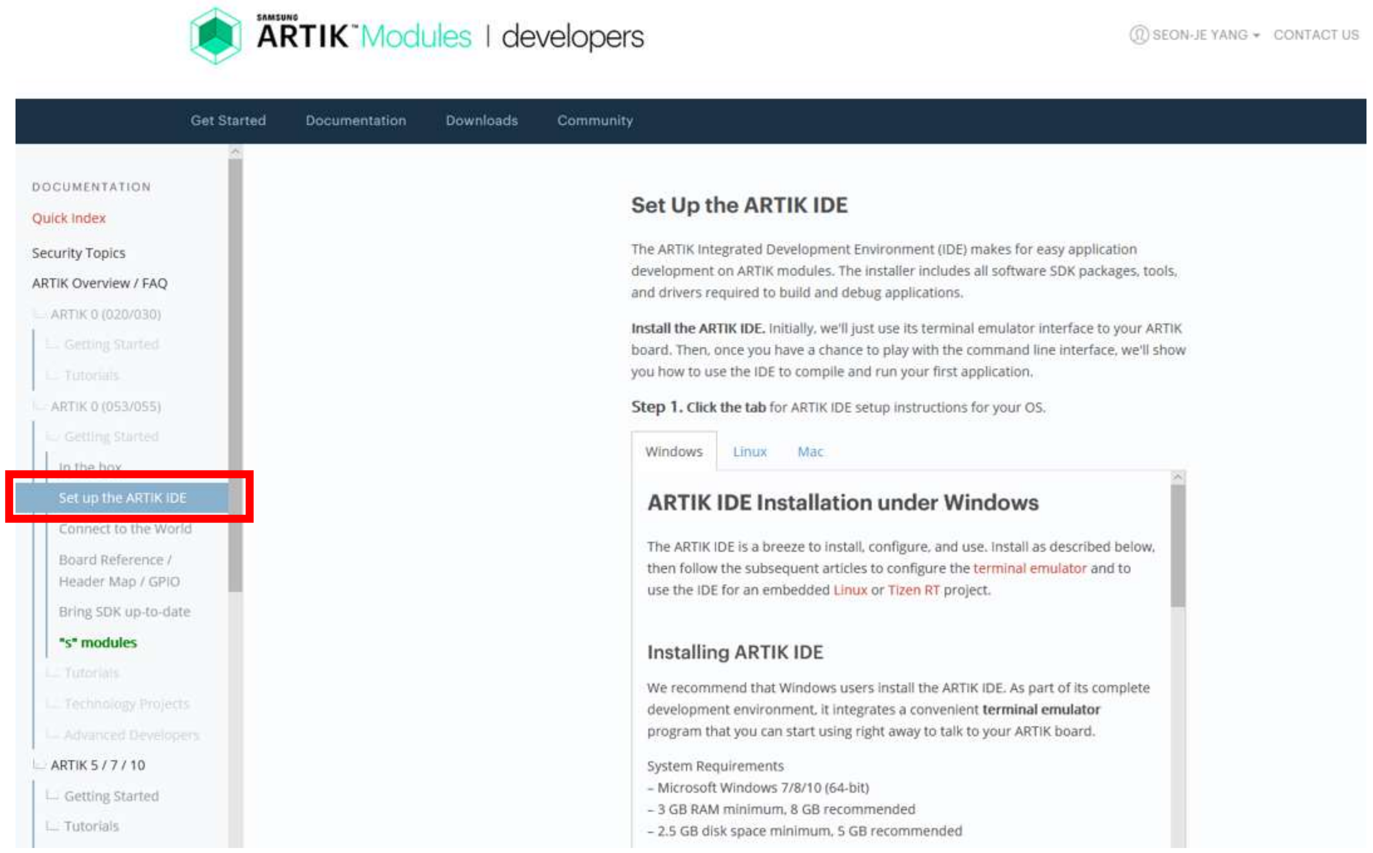
If you'd like help designing, prototyping, or manufacturing, ARTIK Partners can help make your IoT project a success.

Partner directory

ARTIK IDE 설치(2)

❖ [Set up the ARTIK IDE] 클릭

- https://developer.artik.io/documentation/artik-05x/getting-started/prepare-ide.html



Set Up the ARTIK IDE

The ARTIK Integrated Development Environment (IDE) makes for easy application development on ARTIK modules. The installer includes all software SDK packages, tools, and drivers required to build and debug applications.

Install the ARTIK IDE. Initially, we'll just use its terminal emulator interface to your ARTIK board. Then, once you have a chance to play with the command line interface, we'll show you how to use the IDE to compile and run your first application.

Step 1. Click the tab for ARTIK IDE setup instructions for your OS.

ARTIK IDE Installation under Windows

The ARTIK IDE is a breeze to install, configure, and use. Install as described below, then follow the subsequent articles to configure the terminal emulator and to use the IDE for an embedded Linux or Tizen RT project.

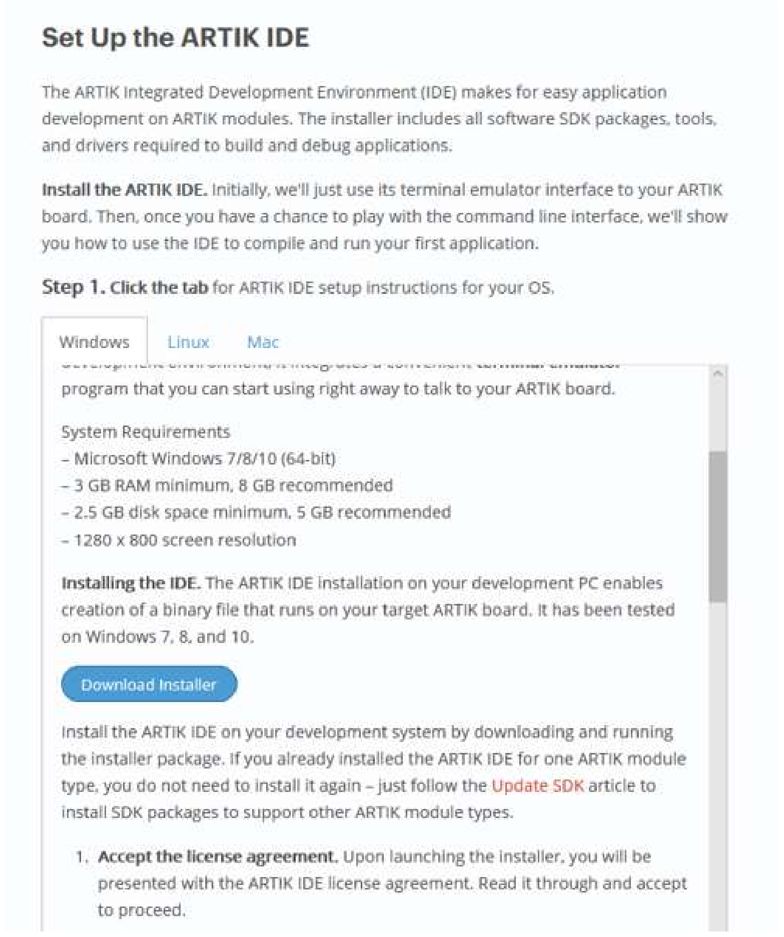
Installing ARTIK IDE

We recommend that Windows users install the ARTIK IDE. As part of its complete development environment, it integrates a convenient **terminal emulator** program that you can start using right away to talk to your ARTIK board.

System Requirements
- Microsoft Windows 7/8/10 (64-bit)
- 3 GB RAM minimum, 8 GB recommended
- 2.5 GB disk space minimum, 5 GB recommended

ARTIK IDE 설치(3)

❖ OS에 맞는 ARTIK IDE 다운로드



Set Up the ARTIK IDE

The ARTIK Integrated Development Environment (IDE) makes for easy application development on ARTIK modules. The installer includes all software SDK packages, tools, and drivers required to build and debug applications.

Install the ARTIK IDE. Initially, we'll just use its terminal emulator interface to your ARTIK board. Then, once you have a chance to play with the command line interface, we'll show you how to use the IDE to compile and run your first application.

Step 1. **Click the tab** for ARTIK IDE setup instructions for your OS.

program that you can start using right away to talk to your ARTIK board.

System Requirements
– Microsoft Windows 7/8/10 (64-bit)
– 3 GB RAM minimum, 8 GB recommended
– 2.5 GB disk space minimum, 5 GB recommended
– 1280 x 800 screen resolution

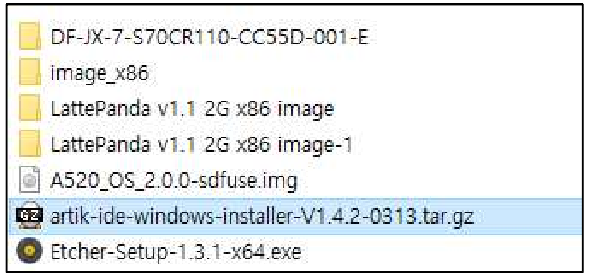
Installing the IDE. The ARTIK IDE installation on your development PC enables creation of a binary file that runs on your target ARTIK board. It has been tested on Windows 7, 8, and 10.

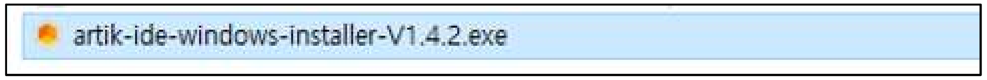
Install the ARTIK IDE on your development system by downloading and running the installer package. If you already installed the ARTIK IDE for one ARTIK module type, you do not need to install it again – just follow the Update SDK article to install SDK packages to support other ARTIK module types.

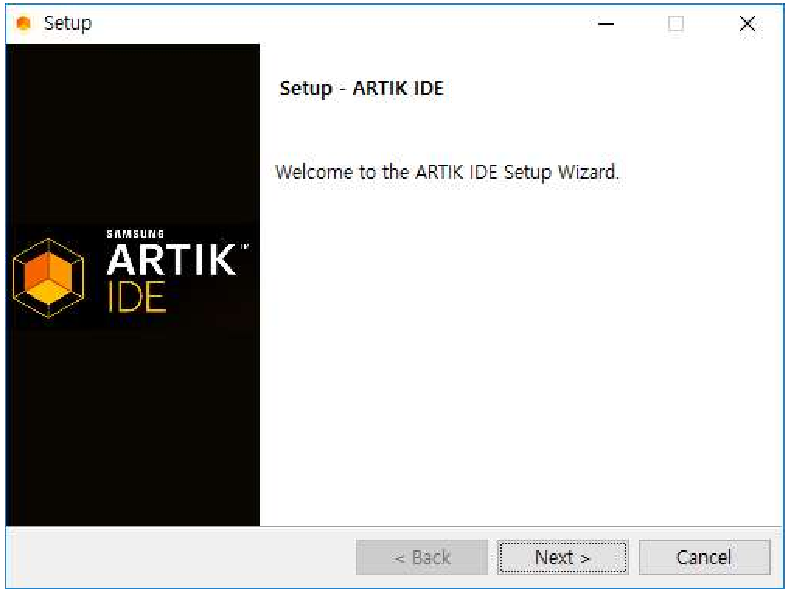
1. **Accept the license agreement.** Upon launching the installer, you will be presented with the ARTIK IDE license agreement. Read it through and accept to proceed.

ARTIK IDE 설치(4)

❖ 다운로드 완료 후 압축 풀고 설치


DF-JX-7-S70CR110-CC55D-001-E
image_x86
LattePanda v1.1 2G x86 image
LattePanda v1.1 2G x86 image-1
A520_OS_2.0.0-sdfuse.img
artik-ide-windows-installer-V1.4.2-0313.tar.gz
Etcher-Setup-1.3.1-x64.exe


artik-ide-windows-installer-V1.4.2.exe


Setup

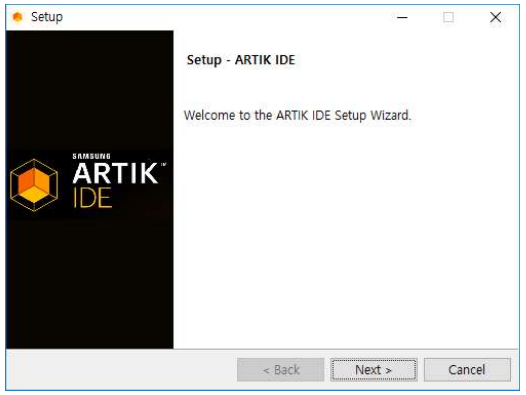
Setup - ARTIK IDE

Welcome to the ARTIK IDE Setup Wizard.

CONTROL N ROBOTICS

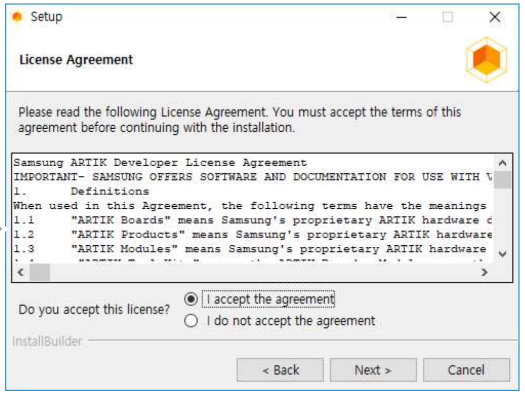
ARTIK IDE 설치(5)

❖ ARTIK IDE 설치


Setup

Setup - ARTIK IDE

Welcome to the ARTIK IDE Setup Wizard.

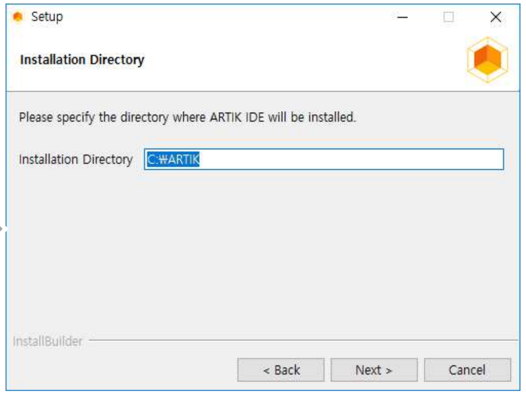

Setup

License Agreement

Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

```
Samsung ARTIK Developer License Agreement
IMPORTANT- SAMSUNG OFFERS SOFTWARE AND DOCUMENTATION FOR USE WITH V
1.      Definitions
When used in this Agreement, the following terms have the meanings
1.1     "ARTIK Boards" means Samsung's proprietary ARTIK hardware c
1.2     "ARTIK Products" means Samsung's proprietary ARTIK hardware
1.3     "ARTIK Modules" means Samsung's proprietary ARTIK hardware
```

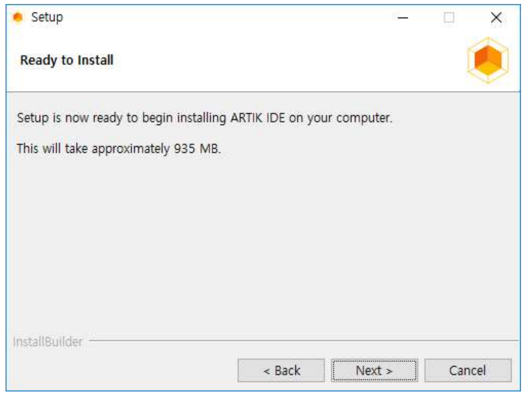
Do you accept this license? [✓ I accept the agreement] I do not accept the agreement


Setup

Installation Directory

Please specify the directory where ARTIK IDE will be installed.

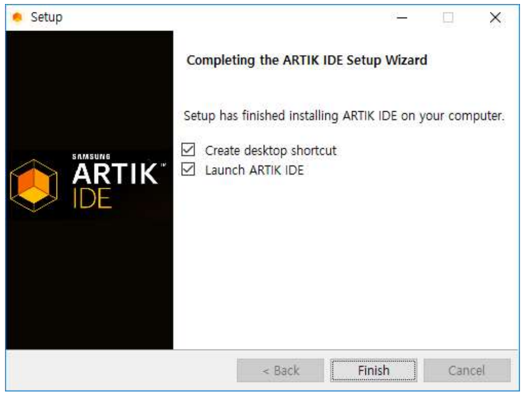
Installation Directory: C:₩ARTIK


Setup

Ready to Install

Setup is now ready to begin installing ARTIK IDE on your computer.

This will take approximately 935 MB.


Setup

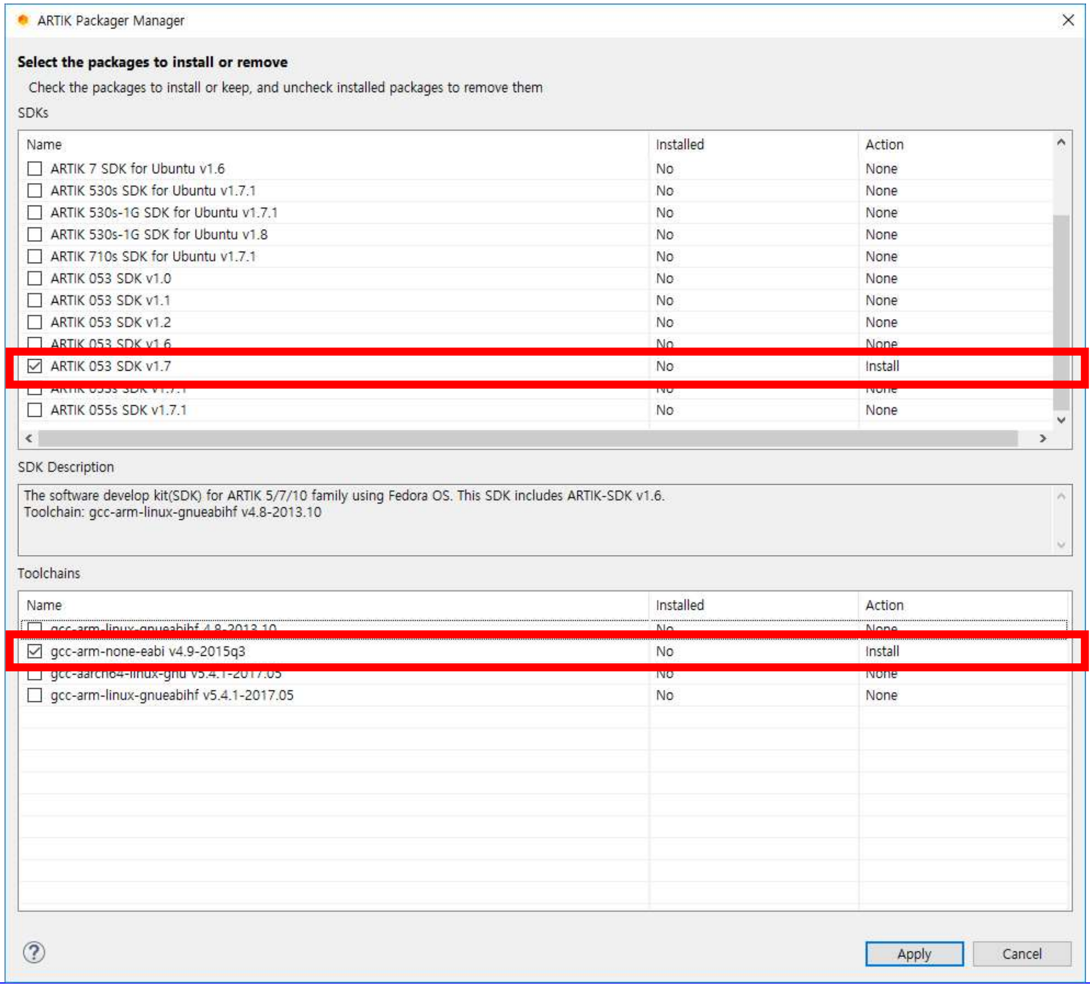
Completing the ARTIK IDE Setup Wizard

Setup has finished installing ARTIK IDE on your computer.

☑ Create desktop shortcut
☑ Launch ARTIK IDE

Package 설치(1)

❖ **ARTIK 053 SDK와 Toolchain 설치**



ARTIK Packager Manager

Select the packages to install or remove

Check the packages to install or keep, and uncheck installed packages to remove them

SDKs

Name	Installed	Action
☐ ARTIK 7 SDK for Ubuntu v1.6	No	None
☐ ARTIK 530s SDK for Ubuntu v1.7.1	No	None
☐ ARTIK 530s-1G SDK for Ubuntu v1.7.1	No	None
☐ ARTIK 530s-1G SDK for Ubuntu v1.8	No	None
☐ ARTIK 710s SDK for Ubuntu v1.7.1	No	None
☐ ARTIK 053 SDK v1.0	No	None
☐ ARTIK 053 SDK v1.1	No	None
☐ ARTIK 053 SDK v1.2	No	None
☐ ARTIK 053 SDK v1.6	No	None
☑ ARTIK 053 SDK v1.7	No	Install
☐ ARTIK 053s SDK v1.7.1	No	None
☐ ARTIK 055s SDK v1.7.1	No	None

SDK Description

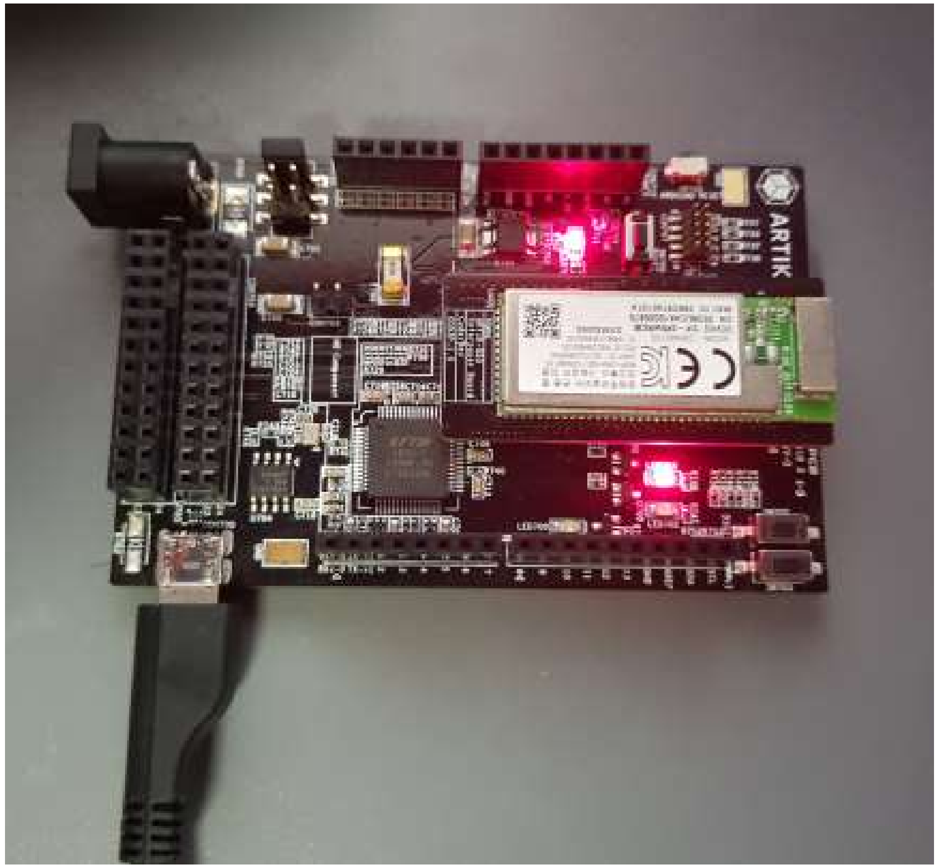
The software develop kit(SDK) for ARTIK 5/7/10 family using Fedora OS. This SDK includes ARTIK-SDK v1.6.
Toolchain: gcc-arm-linux-gnueabihf v4.8-2013.10

Toolchains

Name	Installed	Action
☐ gcc-arm-linux-gnueabihf 4.8-2013.10	No	None
☑ gcc-arm-none-eabi v4.9-2015q3	No	Install
☐ gcc-aarch64-linux-gnu v5.4.1-2017.05	No	None
☐ gcc-arm-linux-gnueabihf v5.4.1-2017.05	No	None

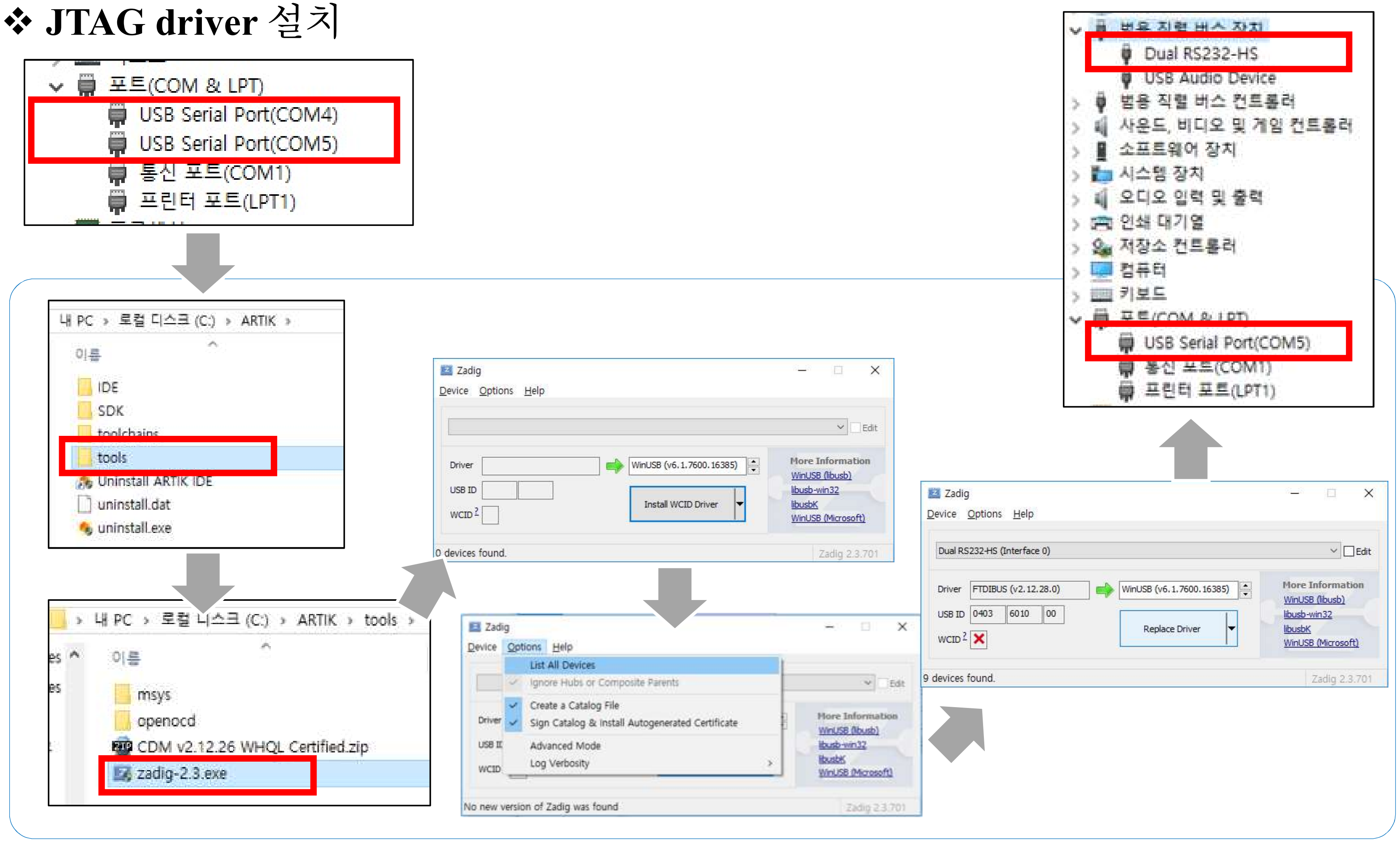
ARTIK 053 Starter Kit

❖ **ARTIK 053 Starter Kit 와 연결 모습**



JTAG 드라이버 설치

❖ JTAG driver 설치



포트(COM & LPT)
USB Serial Port(COM4)
USB Serial Port(COM5)
통신 포트(COM1)
프린터 포트(LPT1)

내 PC › 로컬 디스크 (C:) › ARTIK ›
IDE
SDK
toolchains
tools
Uninstall ARTIK IDE
uninstall.dat
uninstall.exe

내 PC › 로컬 디스크 (C:) › ARTIK › tools ›
msys
openocd
CDM v2.12.26 WHQL Certified.zip
zadig-2.3.exe

Zadig
Driver → WinUSB (v6.1.7600.16385)
0 devices found.
Zadig 2.3.701

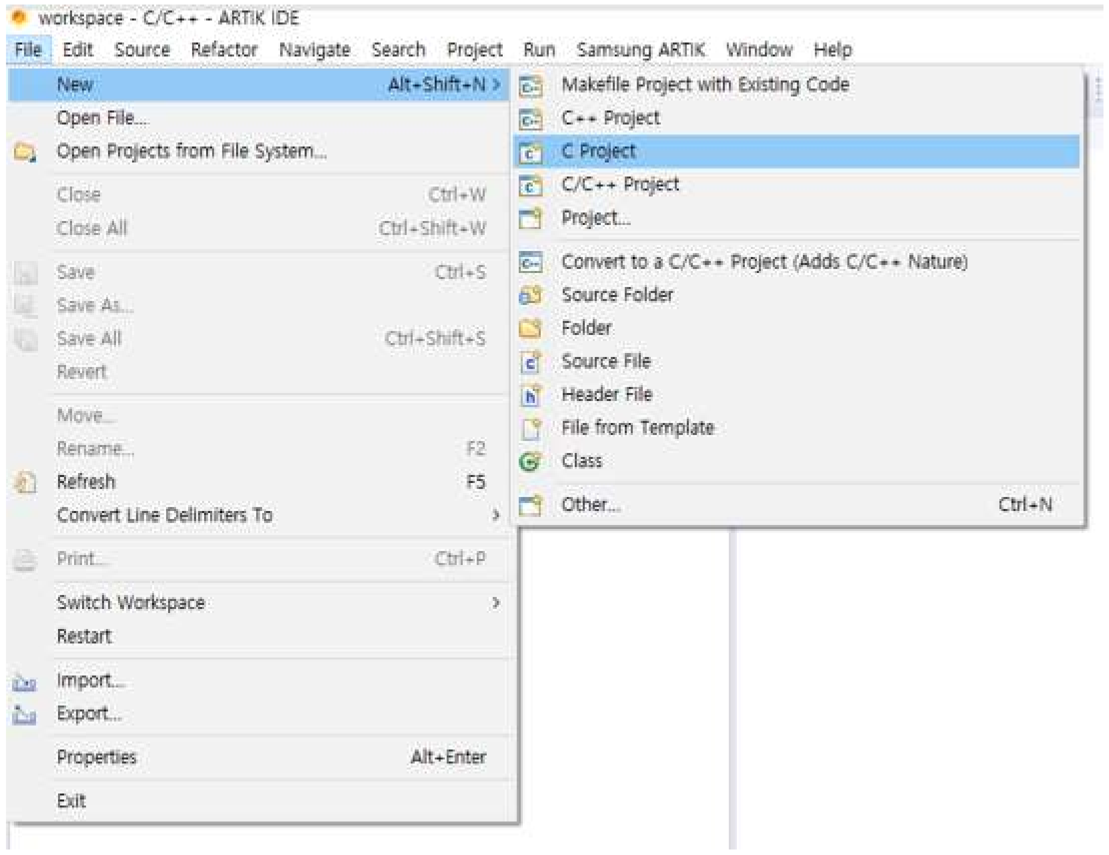
Options: List All Devices; Ignore Hubs or Composite Parents; Create a Catalog File; Sign Catalog & Install Autogenerated Certificate; Advanced Mode; Log Verbosity
No new version of Zadig was found

Zadig
Dual RS232-HS (Interface 0)
Driver FTDIBUS (v2.12.28.0) → WinUSB (v6.1.7600.16385)
USB ID 0403 6010 00
Replace Driver
9 devices found.
Zadig 2.3.701

범용 직렬 버스 장치
Dual RS232-HS
USB Audio Device
범용 직렬 버스 컨트롤러
사운드, 비디오 및 게임 컨트롤러
소프트웨어 장치
시스템 장치
오디오 입력 및 출력
인쇄 대기열
저장소 컨트롤러
컴퓨터
키보드
포트(COM & LPT)
USB Serial Port(COM5)
통신 포트(COM1)
프린터 포트(LPT1)

Make Example Project (1)

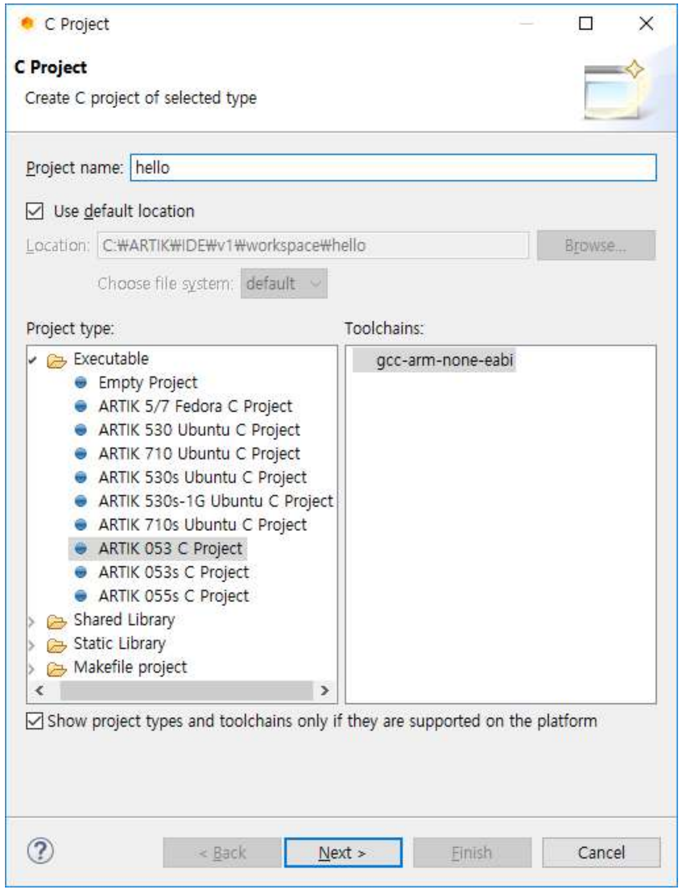
❖ [File] → [New] → [C Project]



Make Example Project (2)

❖ **Type project name**

❖ **Select [ARTIK 053 C Project], [gcc-arm-non-eabi]**



C Project

C Project

Create C project of selected type

Project name: hello

☑ Use default location

Location: C:₩ARTIK₩IDE₩v1₩workspace₩hello

Choose file system: default

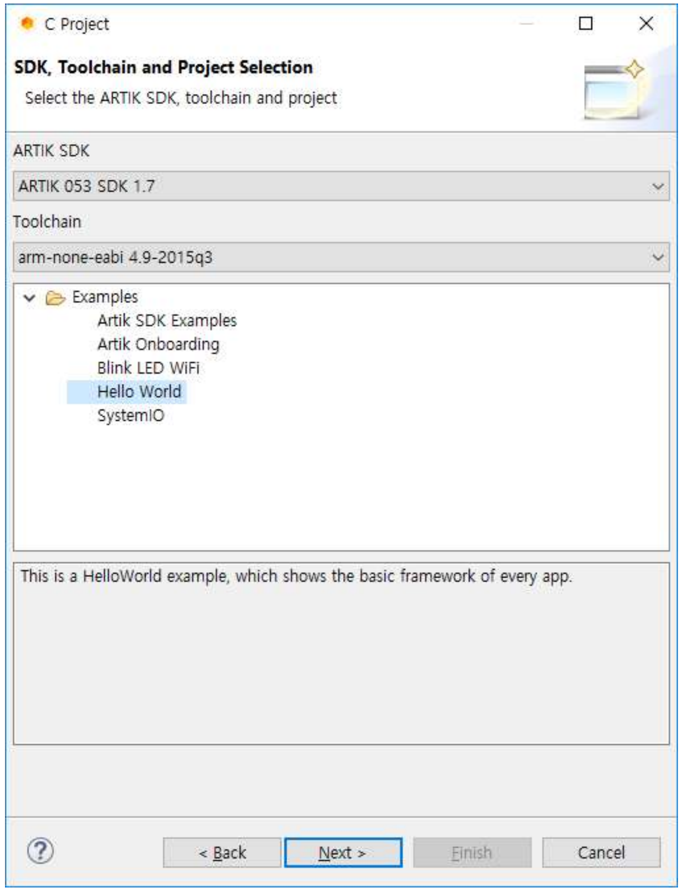
Project type:
- Executable
  - Empty Project
  - ARTIK 5/7 Fedora C Project
  - ARTIK 530 Ubuntu C Project
  - ARTIK 710 Ubuntu C Project
  - ARTIK 530s Ubuntu C Project
  - ARTIK 530s-1G Ubuntu C Project
  - ARTIK 710s Ubuntu C Project
  - ARTIK 053 C Project
  - ARTIK 053s C Project
  - ARTIK 055s C Project
- Shared Library
- Static Library
- Makefile project

Toolchains:
- gcc-arm-none-eabi

☑ Show project types and toolchains only if they are supported on the platform

Make Example Project (3)

- ❖ Select [Examples/Hello World]
- ❖ Change 'Pre-build configuration' option to 'extra'


C Project

SDK, Toolchain and Project Selection

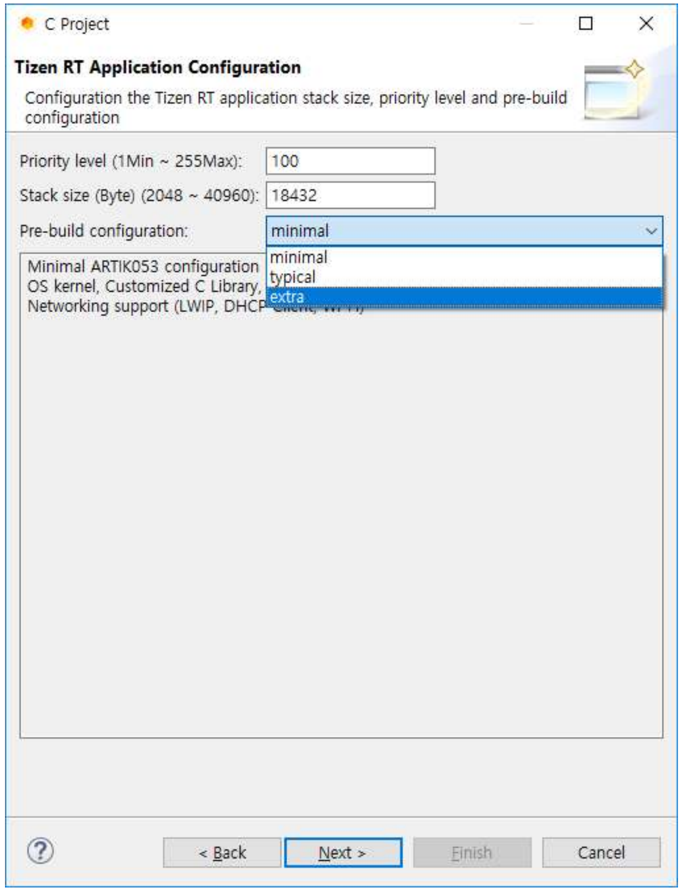
Select the ARTIK SDK, toolchain and project

ARTIK SDK: ARTIK 053 SDK 1.7

Toolchain: arm-none-eabi 4.9-2015q3

- Examples
  - Artik SDK Examples
  - Artik Onboarding
  - Blink LED WiFi
  - Hello World
  - SystemIO

This is a HelloWorld example, which shows the basic framework of every app.


C Project

Tizen RT Application Configuration

Configuration the Tizen RT application stack size, priority level and pre-build configuration

Priority level (1Min ~ 255Max): 100

Stack size (Byte) (2048 ~ 40960): 18432

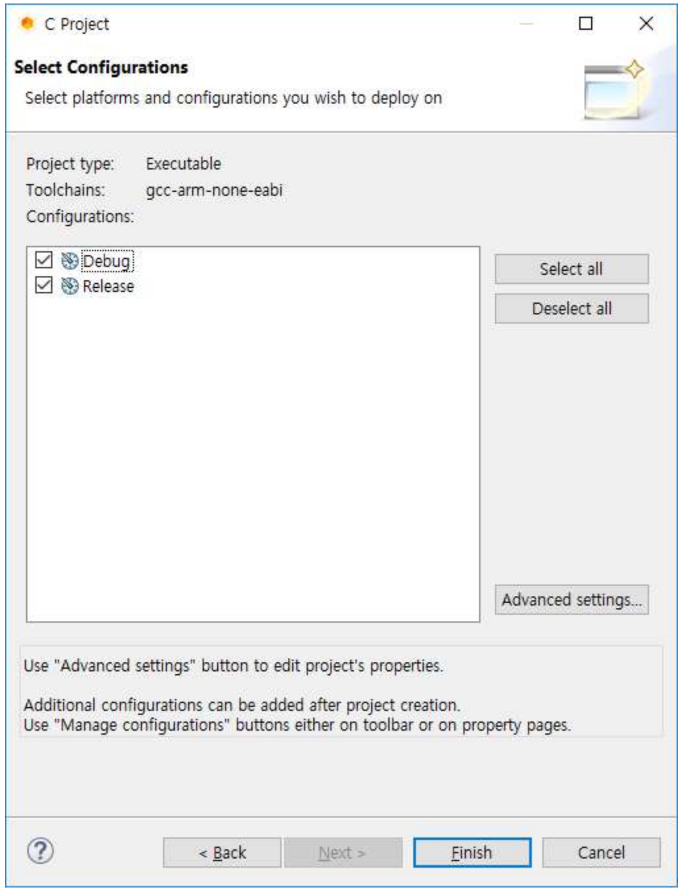
Pre-build configuration: minimal

minimal
typical
extra

Minimal ARTIK053 configuration
OS kernel, Customized C Library,
Networking support (LWIP, DHCP

Make Example Project (4)

❖ **Finish**


C Project

Select Configurations

Select platforms and configurations you wish to deploy on

Project type: Executable
Toolchains: gcc-arm-none-eabi
Configurations:

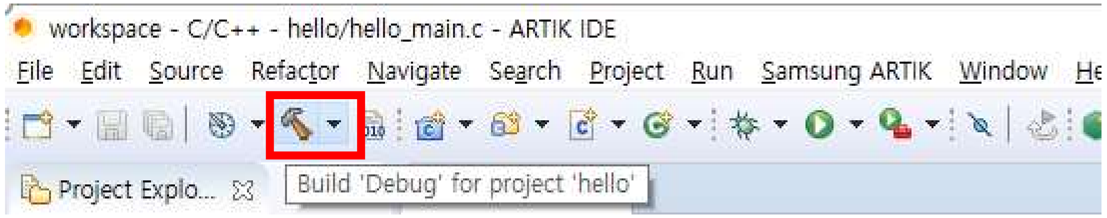
- ☑ Debug
- ☑ Release

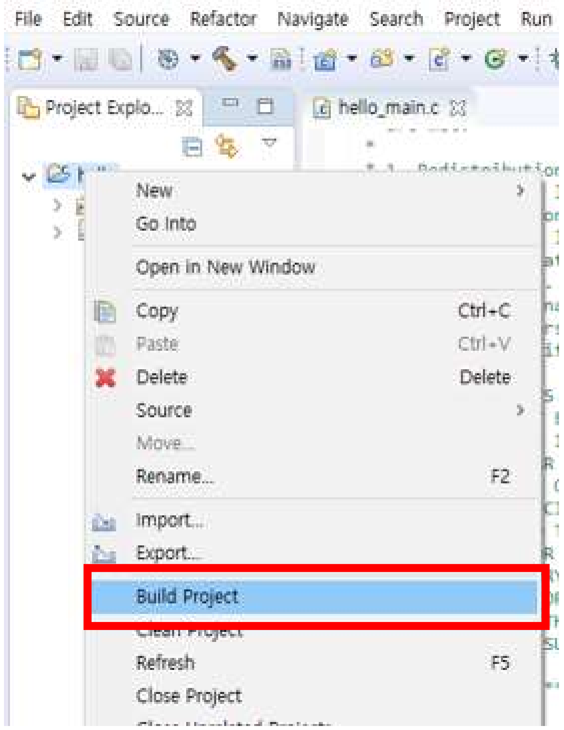
Use "Advanced settings" button to edit project's properties.

Additional configurations can be added after project creation.
Use "Manage configurations" buttons either on toolbar or on property pages.

Build and Flash Example Project (1)

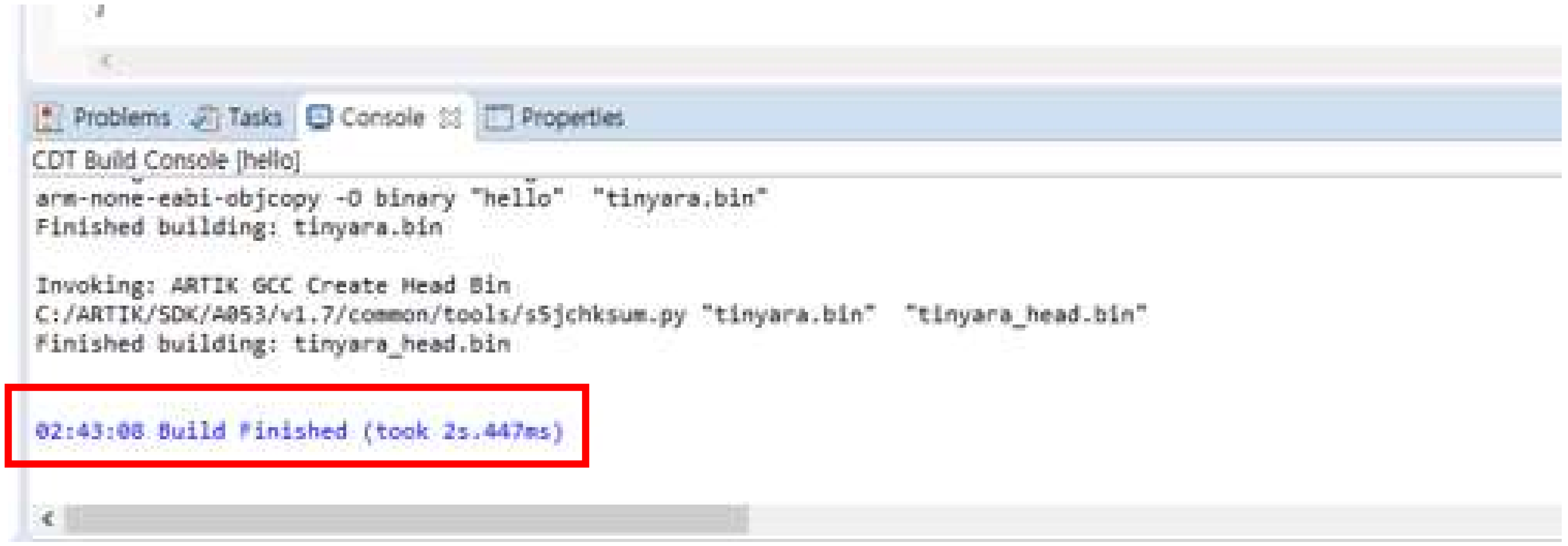
❖ Click the build icon or mouse right click → [Build Project] click


workspace - C/C++ - hello/hello_main.c - ARTIK IDE
File Edit Source Refactor Navigate Search Project Run Samsung ARTIK Window He
Project Explo...
Build 'Debug' for project 'hello'



Build and Flash Example Project (2)

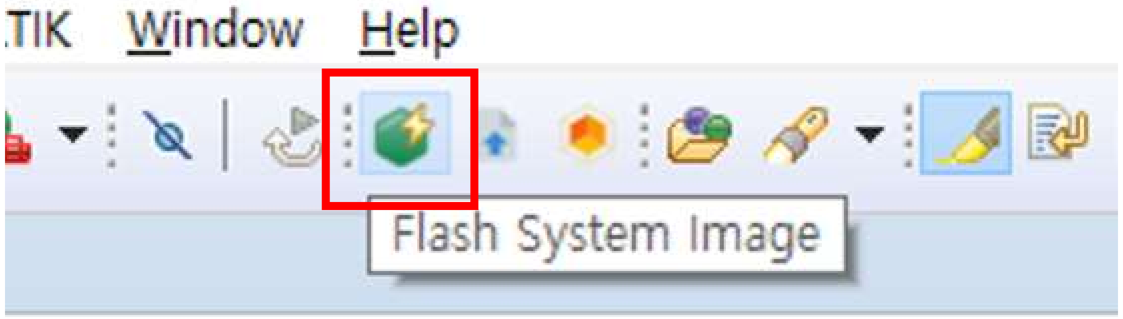
❖ **Finish Build**

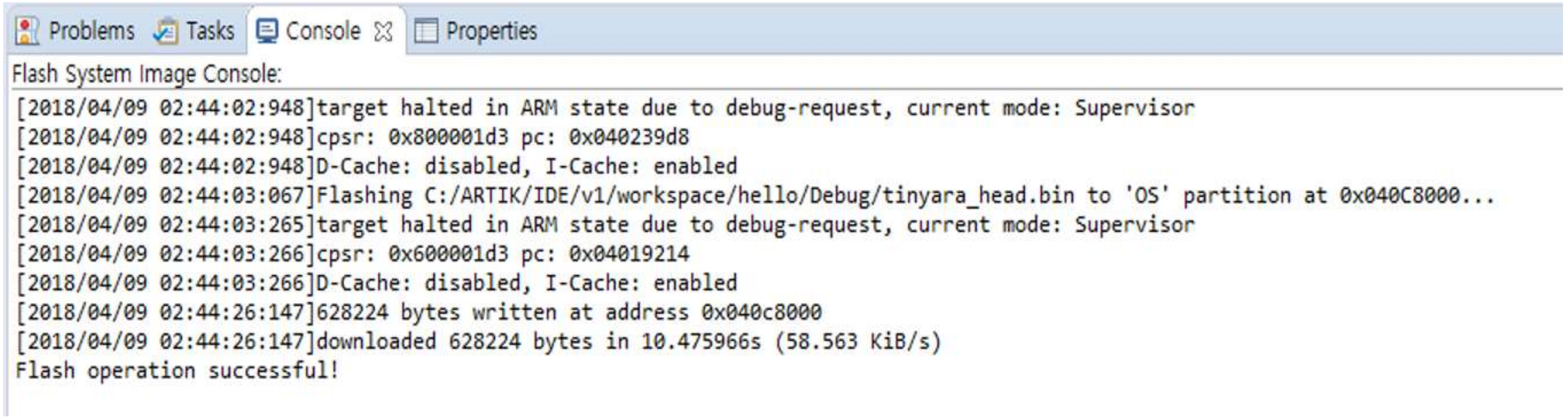


```
arm-none-eabi-objcopy -O binary "hello"  "tinyara.bin"
Finished building: tinyara.bin

Invoking: ARTIK GCC Create Head Bin
C:/ARTIK/SDK/A053/v1.7/common/tools/s5jchksum.py "tinyara.bin"  "tinyara_head.bin"
Finished building: tinyara_head.bin

02:43:08 Build Finished (took 2s.447ms)
```

Build and Flash Example Project (3)

❖ Flash the example code (click the Flash icon)

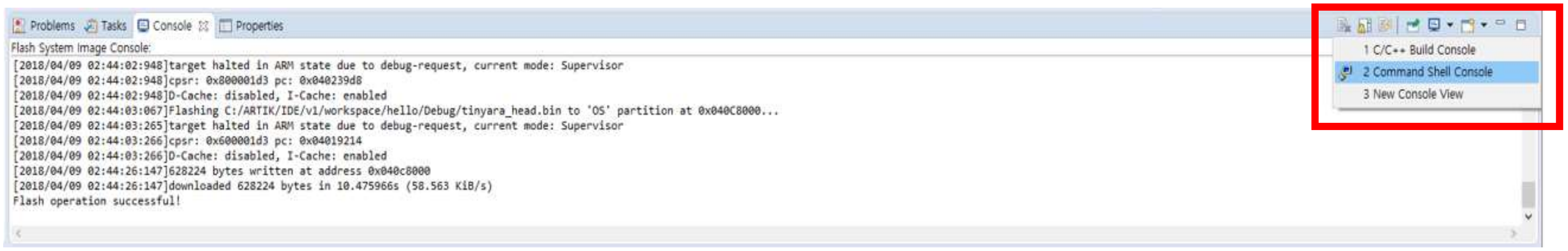




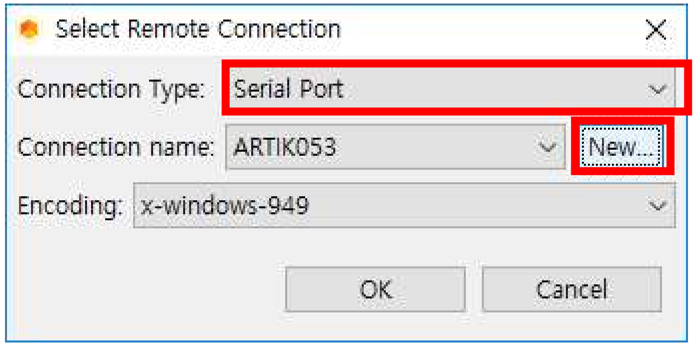
```
Flash System Image Console:
[2018/04/09 02:44:02:948]target halted in ARM state due to debug-request, current mode: Supervisor
[2018/04/09 02:44:02:948]cpsr: 0x800001d3 pc: 0x040239d8
[2018/04/09 02:44:02:948]D-Cache: disabled, I-Cache: enabled
[2018/04/09 02:44:03:067]Flashing C:/ARTIK/IDE/v1/workspace/hello/Debug/tinyara_head.bin to 'OS' partition at 0x040C8000...
[2018/04/09 02:44:03:265]target halted in ARM state due to debug-request, current mode: Supervisor
[2018/04/09 02:44:03:266]cpsr: 0x600001d3 pc: 0x04019214
[2018/04/09 02:44:03:266]D-Cache: disabled, I-Cache: enabled
[2018/04/09 02:44:26:147]628224 bytes written at address 0x040c8000
[2018/04/09 02:44:26:147]downloaded 628224 bytes in 10.475966s (58.563 KiB/s)
Flash operation successful!
```

Connection

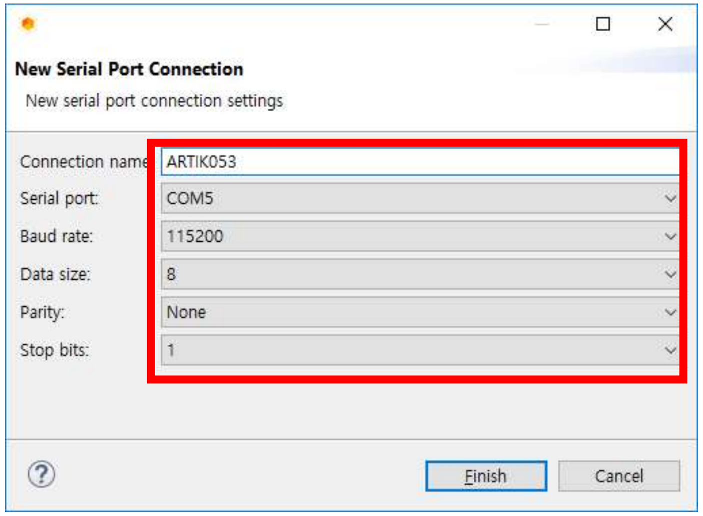
❖ Make Console Window (≈putty)



```
Flash System Image Console:
[2018/04/09 02:44:02:948]target halted in ARM state due to debug-request, current mode: Supervisor
[2018/04/09 02:44:02:948]cpsr: 0x800001d3 pc: 0x040239d8
[2018/04/09 02:44:02:948]D-Cache: disabled, I-Cache: enabled
[2018/04/09 02:44:03:067]Flashing C:/ARTIK/IDE/v1/workspace/hello/Debug/tinyara_head.bin to 'OS' partition at 0x040C8000...
[2018/04/09 02:44:03:265]target halted in ARM state due to debug-request, current mode: Supervisor
[2018/04/09 02:44:03:266]cpsr: 0x600001d3 pc: 0x04019214
[2018/04/09 02:44:03:266]D-Cache: disabled, I-Cache: enabled
[2018/04/09 02:44:26:147]628224 bytes written at address 0x040c8000
[2018/04/09 02:44:26:147]downloaded 628224 bytes in 10.475966s (58.563 KiB/s)
Flash operation successful!
```



Connection Type: Serial Port
Connection name: ARTIK053
Encoding: x-windows-949



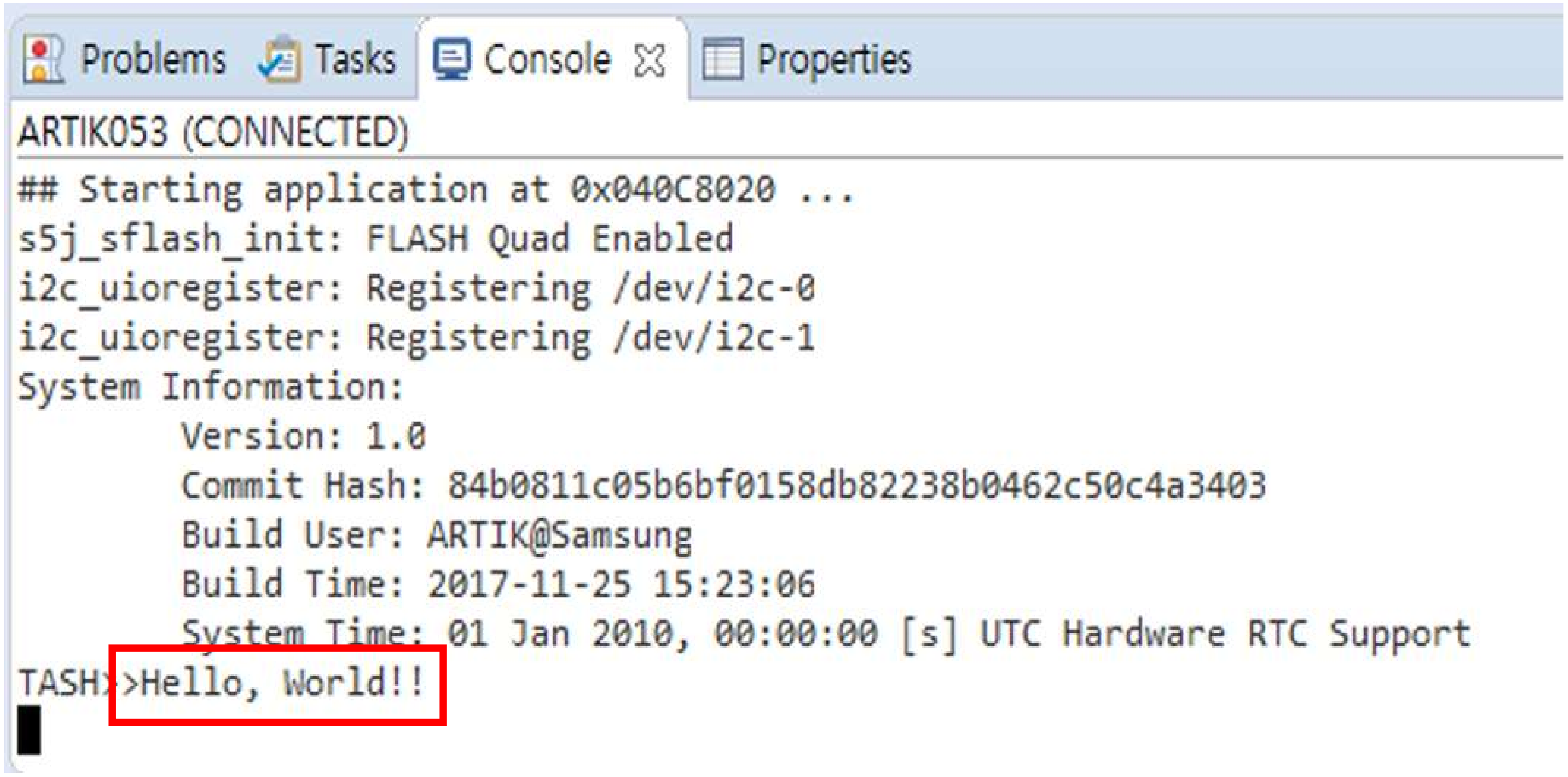
New Serial Port Connection

New serial port connection settings

Connection name	ARTIK053
Serial port:	COM5
Baud rate:	115200
Data size:	8
Parity:	None
Stop bits:	1

Results

❖ **ARTIK 053 said 'Hello, World!'**



```
ARTIK053 (CONNECTED)
## Starting application at 0x040C8020 ...
s5j_sflash_init: FLASH Quad Enabled
i2c_uioregister: Registering /dev/i2c-0
i2c_uioregister: Registering /dev/i2c-1
System Information:
        Version: 1.0
        Commit Hash: 84b0811c05b6bf0158db82238b0462c50c4a3403
        Build User: ARTIK@Samsung
        Build Time: 2017-11-25 15:23:06
        System Time: 01 Jan 2010, 00:00:00 [s] UTC Hardware RTC Support
TASH>>Hello, World!!
```

기본 예제

- Hello World!
- LED
- Switch
- PWM
- ADC

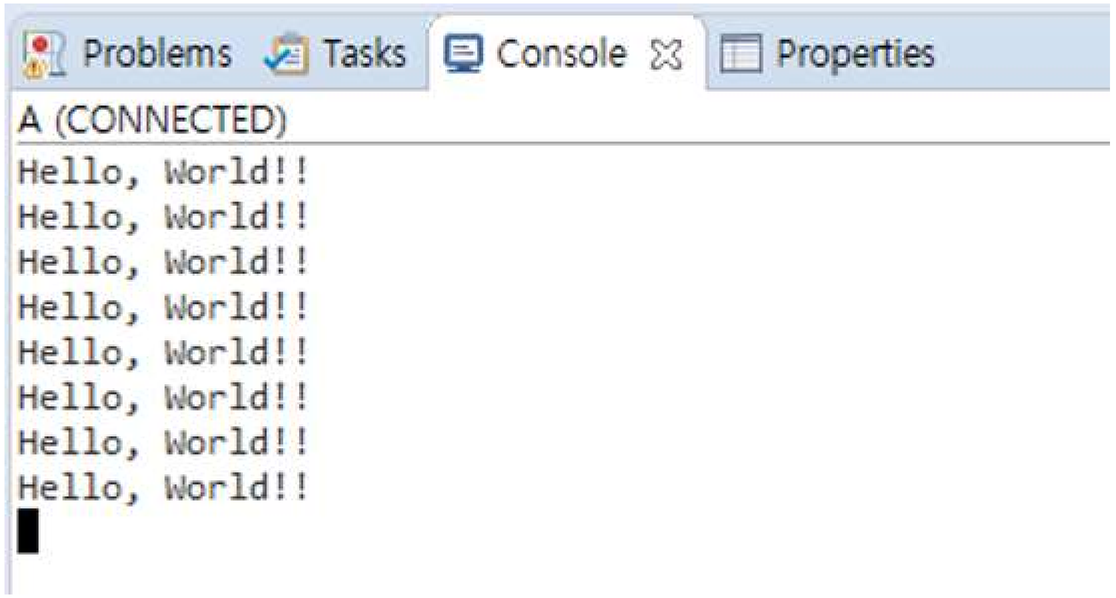
Hello World! (1)

❖ main_hello.c

```c
#include <stdio.h>

void main(void)
{
		while(1)
		{
				printf("Hello, World!!\n");
				up_mdelay(1000);
		}
}
```

Hello World! (2)

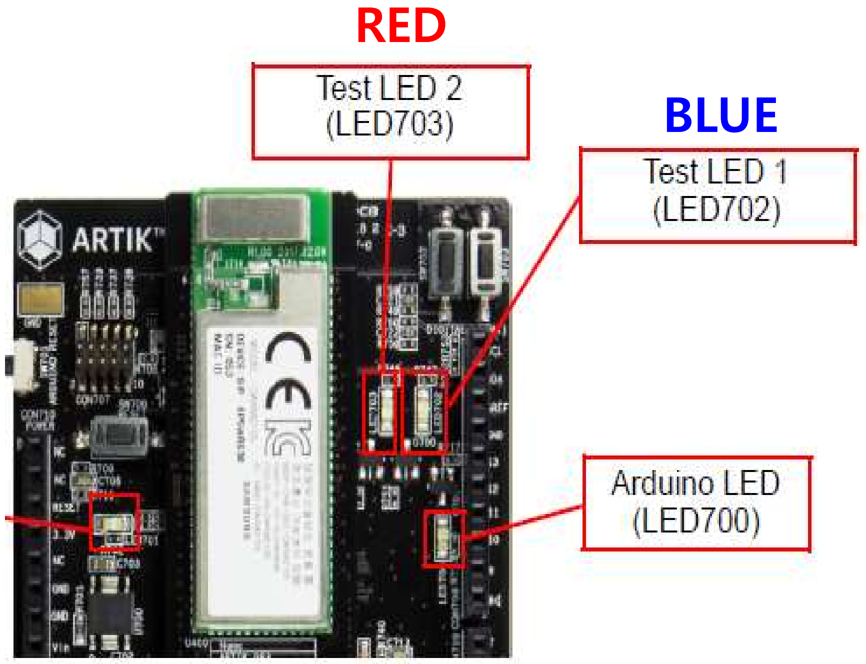
❖ **Result**

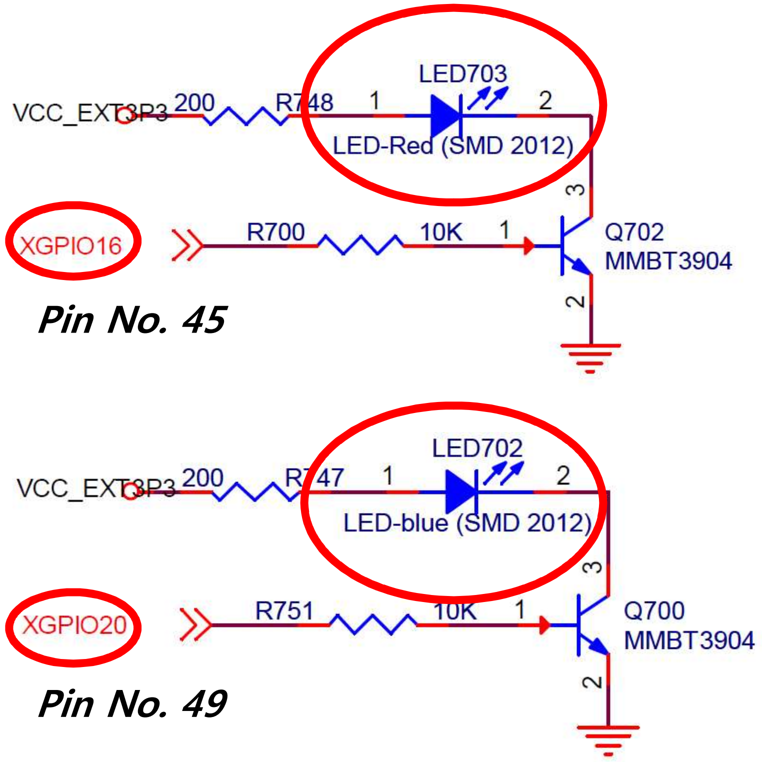


```
A (CONNECTED)
Hello, World!!
Hello, World!!
Hello, World!!
Hello, World!!
Hello, World!!
Hello, World!!
Hello, World!!
Hello, World!!
```

LED (1)

❖ **LEDs on ARTIK 053**





LED (2)

❖ gpio.h

```
#ifndef GPIO_H_
#define GPIO_H_

#include <fcntl.h>
#include <tinyara/gpio.h>

#define HIGH 1
#define LOW 0
void gpio_write(int port, int value);
int gpio_read(int port);

#endif /*GPIO_H_*/
```

LED (3)

❖ gpio.c

```
#include "gpio.h"

void gpio_write(int port, int value)
{
            char str[4];
            static char devpath[16];
            snprintf(devpath, 16, "/dev/gpio%d", port);
            int fd = open(devpath, O_RDWR);

            ioctl(fd, GPIOIOC_SET_DIRECTION, GPIO_DIRECTION_OUT);
            write(fd, str, snprintf(str, 4, "%d", value != 0) + 1);

            close(fd);
}

int gpio_read(int port)
{
            char buf[4];
            char devpath[16];
            snprintf(devpath, 16, "/dev/gpio%d", port);
            int fd = open(devpath, O_RDWR);

            read(fd, buf, sizeof(buf));

            close(fd);

            return buf[0]=='1';
}
```

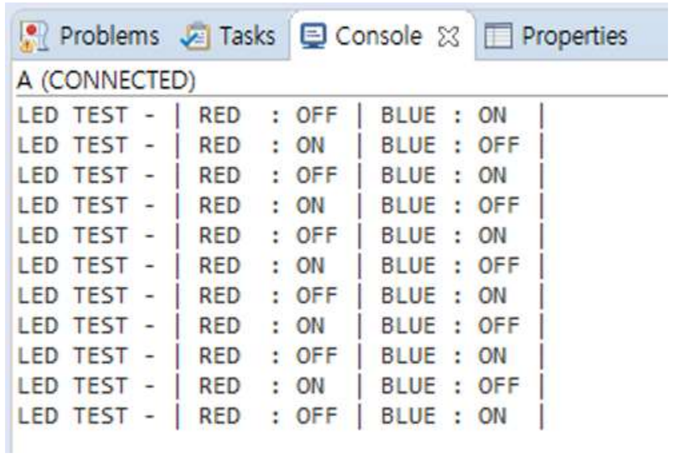
LED (4)

❖ **main_led.c**

```c
#include <stdio.h>
#include "gpio.h"

#define LED_RED 45
#define LED_BLUE 49

void main(void)
{
       while(1)
       {
             printf("LED TEST - | RED  : ON  | BLUE : OFF | \n");
             gpio_write(LED_RED,HIGH);
             gpio_write(LED_BLUE,LOW);
             up_mdelay(1000);

             printf("LED TEST - | RED  : OFF | BLUE : ON  | \n");
             gpio_write(LED_RED,LOW);
             gpio_write(LED_BLUE,HIGH);
             up_mdelay(1000);
       }
}
```

LED (5)

❖ results

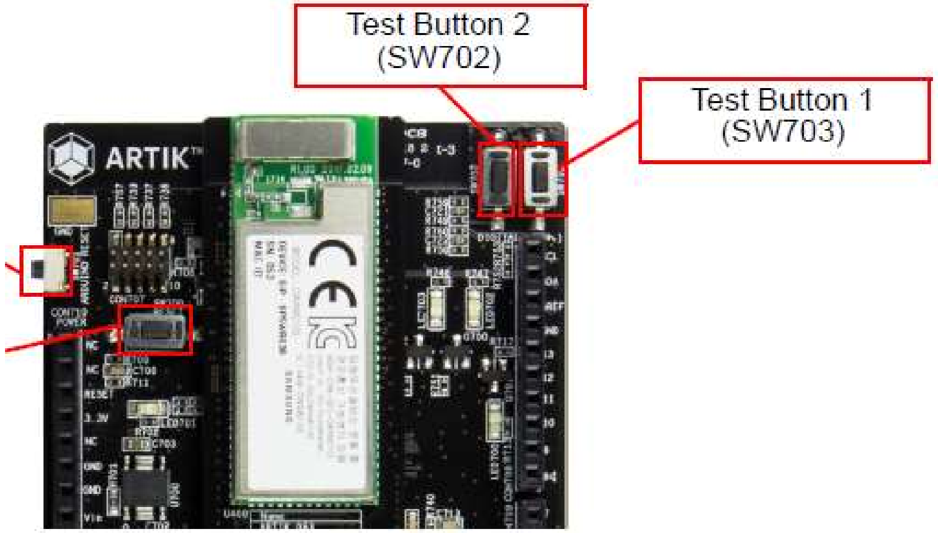


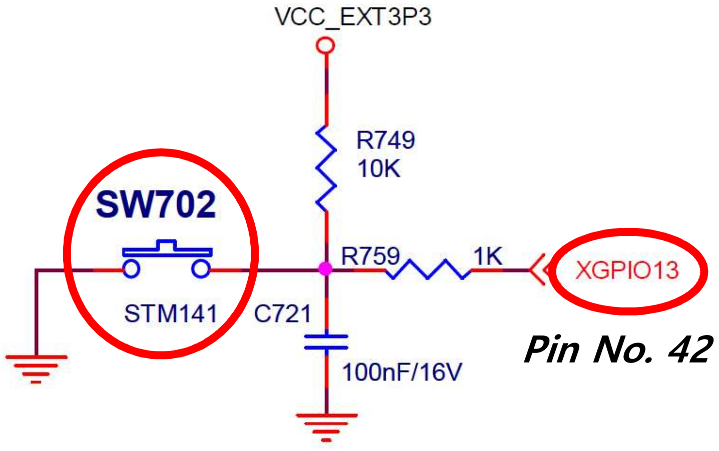
Problems | Tasks | Console | Properties

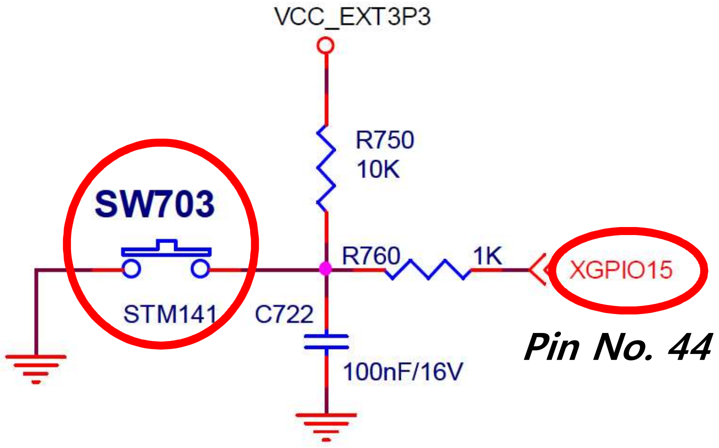
```
A (CONNECTED)
LED TEST - | RED  : OFF | BLUE : ON  |
LED TEST - | RED  : ON  | BLUE : OFF |
LED TEST - | RED  : OFF | BLUE : ON  |
LED TEST - | RED  : ON  | BLUE : OFF |
LED TEST - | RED  : OFF | BLUE : ON  |
LED TEST - | RED  : ON  | BLUE : OFF |
LED TEST - | RED  : OFF | BLUE : ON  |
LED TEST - | RED  : ON  | BLUE : OFF |
LED TEST - | RED  : OFF | BLUE : ON  |
LED TEST - | RED  : ON  | BLUE : OFF |
LED TEST - | RED  : OFF | BLUE : ON  |
```

Switch (1)

❖ **Switches on ARTIK 053**







Switch (2)

❖ main_switch.c (1)

```c
#include <stdio.h>
#include "gpio.h"

//GPIO 13(42) 0/1 : switch on/off
//GPIO 16(45) 0/1 : RED off/on
//GPIO 15(44) 0/1 : switch on/off
//GPIO 20(49) 0/1 : BLUE off/on

#define LED_RED 45
#define LED_BLUE 49
#define SW_RED 42
#define SW_BLUE 44

void main(void)
{
		int sw_red_val = 0;
		int sw_blue_val = 0;

		while(1)
		{
				sw_red_val = gpio_read(SW_RED);
				sw_blue_val = gpio_read(SW_BLUE);

				printf("Read GPIO [RED SW(%d) : %d, BLUE SW(%d) : %d]\n", SW_RED,sw_red_val,SW_BLUE,sw_blue_val);
```

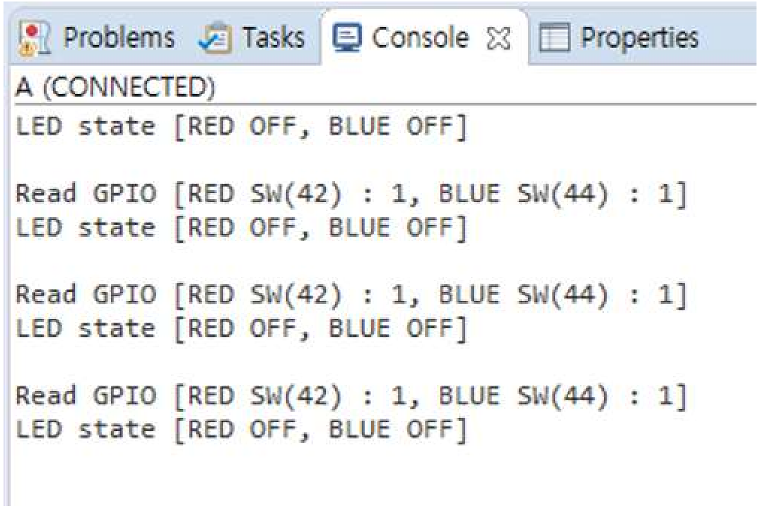
Switch (3)

❖ **main_switch.c (2)**

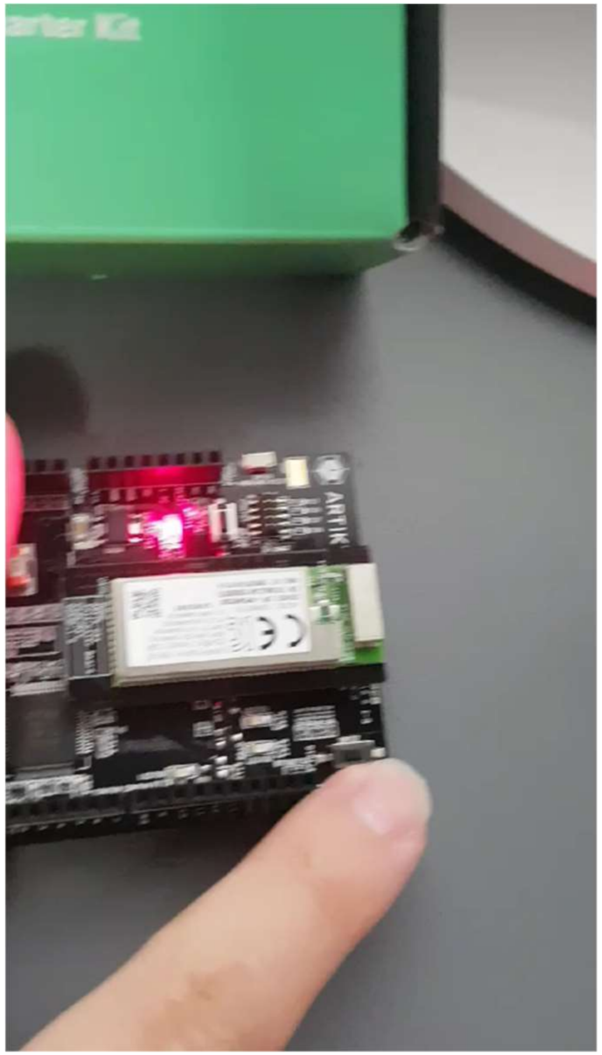
```
                        if (sw_red_val == 0)
                        {
                                    gpio_write(LED_RED,HIGH);
                                    printf("LED state [RED  ON, ");
                        }
                        else
                        {
                                    gpio_write(LED_RED,LOW);
                                    printf("LED state [RED OFF, ");
                        }

                        if (sw_blue_val == 0)
                        {
                                    gpio_write(LED_BLUE,HIGH);
                                    printf("BLUE  ON]\n\n");
                        }
                        else
                        {
                                    gpio_write(LED_BLUE,LOW);
                                    printf("BLUE OFF]\n\n");
                        }

                        up_mdelay(1000);
            }
}
```

Switch (4)

❖ Results

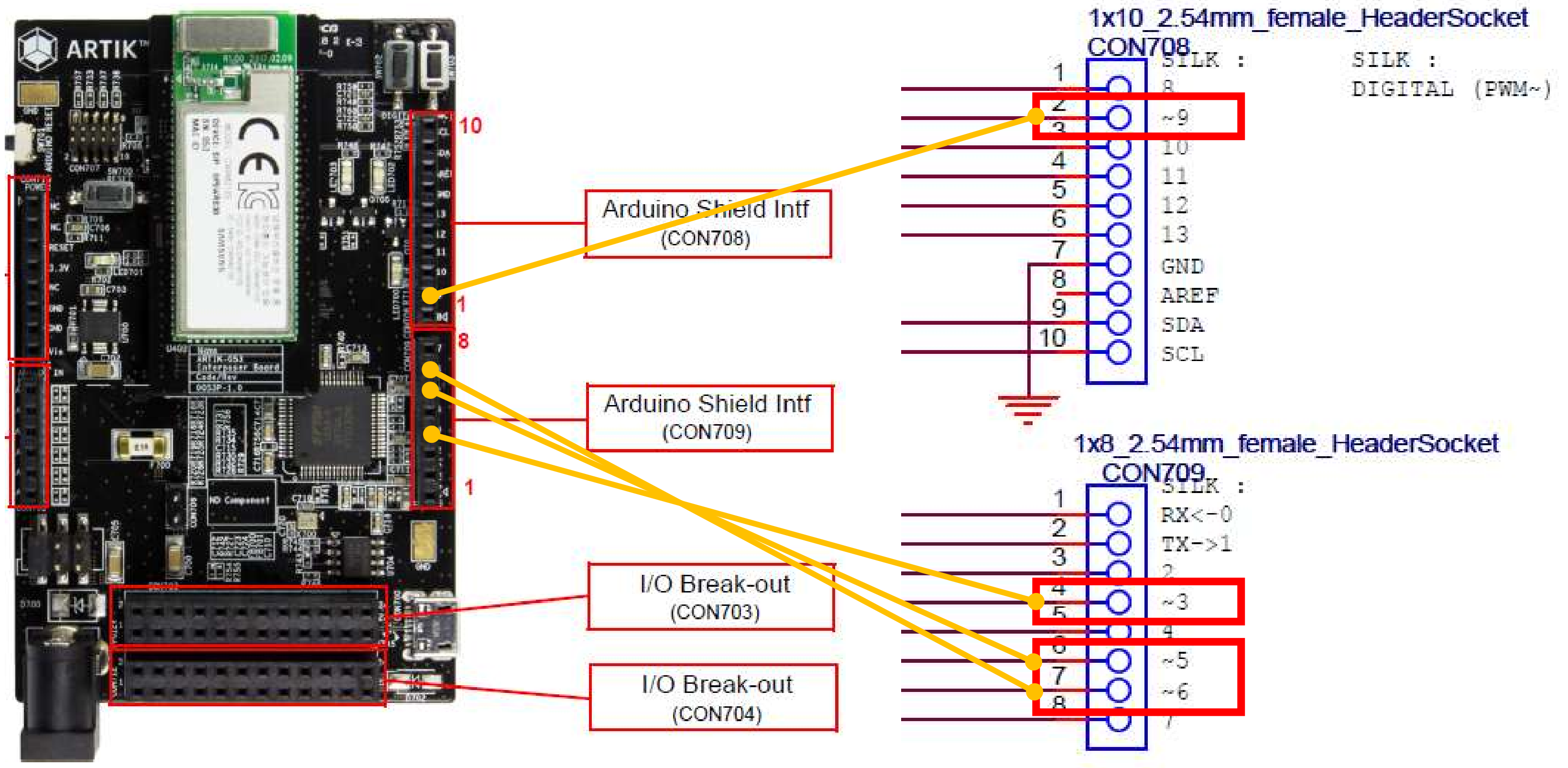


```
A (CONNECTED)
LED state [RED OFF, BLUE OFF]

Read GPIO [RED SW(42) : 1, BLUE SW(44) : 1]
LED state [RED OFF, BLUE OFF]

Read GPIO [RED SW(42) : 1, BLUE SW(44) : 1]
LED state [RED OFF, BLUE OFF]

Read GPIO [RED SW(42) : 1, BLUE SW(44) : 1]
LED state [RED OFF, BLUE OFF]
```



PWM (1)

❖ **PWMs in ARTIK 053**



PWM (2)

❖ pwm.h

```
#ifndef PWM_H_
#define PWM_H_

#include <fcntl.h>
#include <tinyara/pwm.h>

#define ENABLE 1
#define DISABLE 0

int pwm_open(int port);
void pwm_write(int fd, int period, int duty_cycle);
void pwm_close(int fd);

#endif /*PWM_H_*/
```

PWM (3)

❖ pwm.c (1)

```
#include "PWM.h"

int pwm_open(int port)
{
            int fd;

            if (port==0) fd=open("/dev/pwm0",O_RDWR);
            else if (port==1) fd=open("/dev/pwm1",O_RDWR);
            else if (port==2) fd=open("/dev/pwm2",O_RDWR);
            else if (port==3) fd=open("/dev/pwm3",O_RDWR);
            else if (port==4) fd=open("/dev/pwm4",O_RDWR);
            else if (port==5) fd=open("/dev/pwm5",O_RDWR);

            return fd;
}
```

PWM (4)

❖ pwm.c (2)

```
void pwm_write(int fd, int period, int duty_cycle)
{
            int frequency;
            ub16_t duty;
            struct pwm_info_s pwm_info;

            //set pwm_info parameter
            frequency = 1000000 / period;
            duty = duty_cycle * 65536 / period;
            pwm_info.frequency = frequency;
            pwm_info.duty = duty;

            ioctl(fd, PWMIOC_SETCHARACTERISTICS, (unsigned long)((uintptr_t)&pwm_info));
            ioctl(fd, PWMIOC_START);
}

void pwm_close(int fd)
{
            ioctl(fd, PWMIOC_STOP);
            close(fd);
}
```

CONTROL ROBOTICS

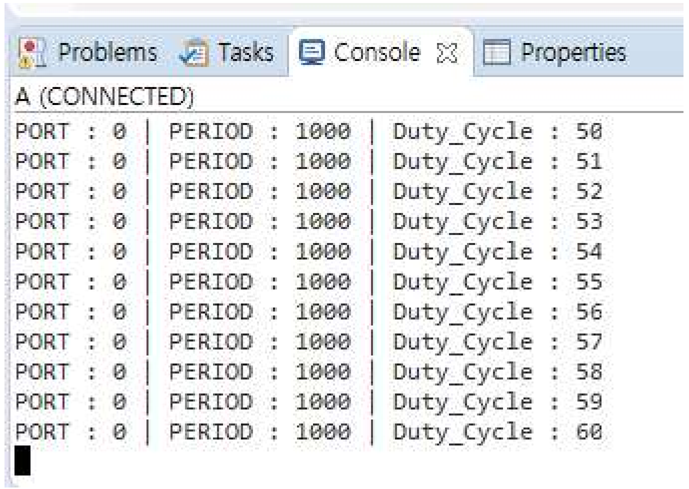
PWM (5)

❖ main_pwm.c

```
#include <stdio.h>
#include "pwm.h"

#define PWM_PIN 0
#define PERIOD 1000

void main(void)
{
            int i;
            int fd;

            fd=pwm_open(PWM_PIN);
            while(1)
            {
                        for (i=0 ; i< 1000 ; i=i+10)
                        {
                                    printf("PORT : %d | PERIOD : %d | Duty_Cycle : %d\n", PWM_PIN, PERIOD, i/10);
                                    pwm_write(fd, PERIOD, i);
                                    up_mdelay(100);
                        }

                        for (i=990 ; i>0 ; i=i-10)
                        {
                                    printf("PORT : %d | PERIOD : %d | Duty_Cycle : %d \n", PWM_PIN, PERIOD, i/10);
                                    pwm_write(fd, PERIOD, i);
                                    up_mdelay(100);
                        }
            }
            pwm_close(fd);
}
```

PWM (6)

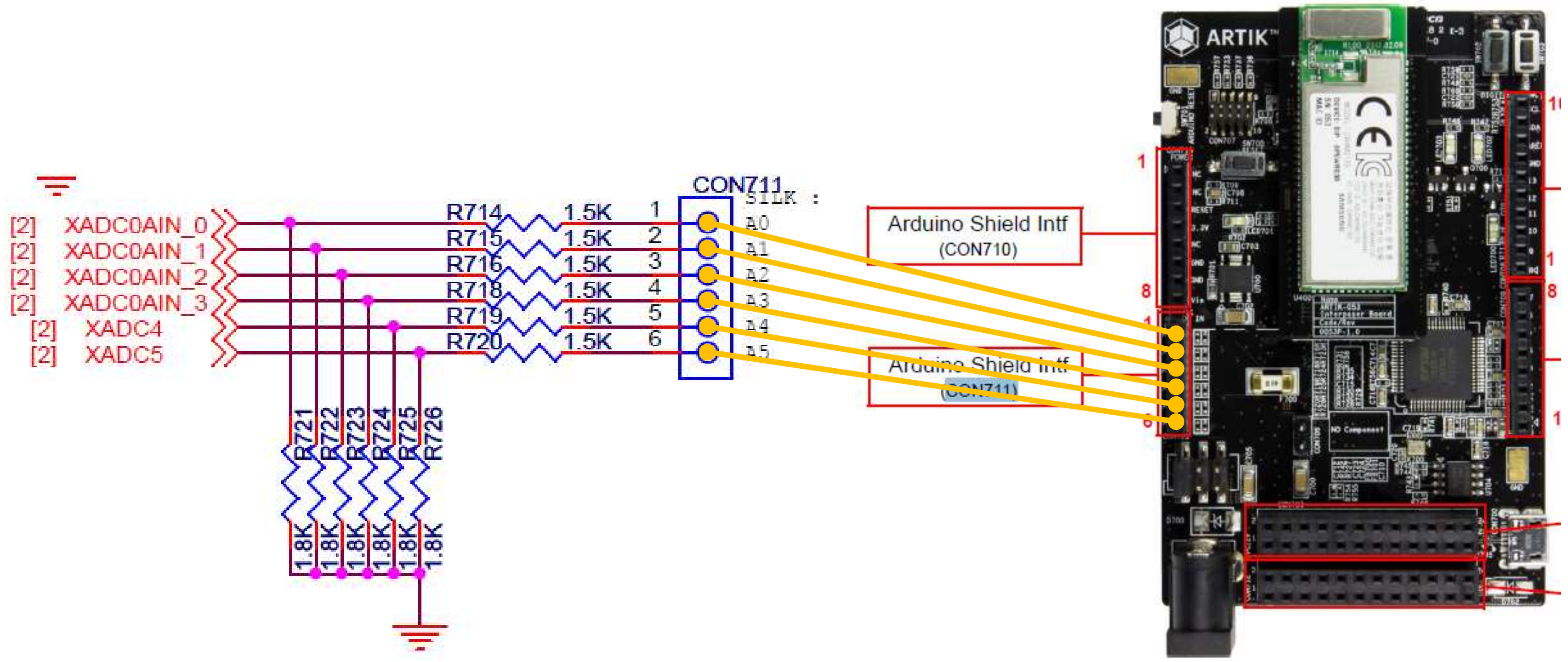
❖ **Results**



```
A (CONNECTED)
PORT : 0 | PERIOD : 1000 | Duty_Cycle : 50
PORT : 0 | PERIOD : 1000 | Duty_Cycle : 51
PORT : 0 | PERIOD : 1000 | Duty_Cycle : 52
PORT : 0 | PERIOD : 1000 | Duty_Cycle : 53
PORT : 0 | PERIOD : 1000 | Duty_Cycle : 54
PORT : 0 | PERIOD : 1000 | Duty_Cycle : 55
PORT : 0 | PERIOD : 1000 | Duty_Cycle : 56
PORT : 0 | PERIOD : 1000 | Duty_Cycle : 57
PORT : 0 | PERIOD : 1000 | Duty_Cycle : 58
PORT : 0 | PERIOD : 1000 | Duty_Cycle : 59
PORT : 0 | PERIOD : 1000 | Duty_Cycle : 60
```

ADC (1)

❖ **ADCs in ARTIK 053**



ADC (2)

❖ adc.h

```
#ifndef ADC_H_
#define ADC_H_

#include <errno.h>
#include <fcntl.h>
#include <tinyara/analog/adc.h>
#include <tinyara/analog/ioctl.h>

#define S5J_ADC_MAX_CHANNELS 4

int read_adc(int channel);

#endif /*ADC_H_*/
```

ADC (3)

❖ adc.c (1)

```
#include "adc.h"

int read_adc(int channel)
{
             int fd, ret;
             struct adc_msg_s sample[S5J_ADC_MAX_CHANNELS];
             int32_t data;
             size_t readsize;
             ssize_t nbytes;

             fd = open("/dev/adc0", O_RDONLY);

             if(fd<0)
             {
                          printf("%s : open failed : %d \n",__func__,errno);
                          return -1;
             }
```

ADC (4)

❖ adc.c (2)

```
	for(;;)
	{
		ret = ioctl(fd,ANIOC_TRIGGER, 0);
		if (ret<0)
		{
				printf("%s : ioctl failed : %d \n",__func__,errno);
				close(fd);
				return -1;
		}
		readsize = S5J_ADC_MAX_CHANNELS * sizeof(struct adc_msg_s);
		nbytes = read(fd, sample, readsize);

		if(nbytes <0)
		{
				if(errno!=EINTR)
				{
						printf("%s : read failed : %d \n",__func__,errno);
						close(fd);
						return -1;
				}
		}
		else if (nbytes==0)
		{
				printf("%s : No data read, Ignoring\n", __func__);
		}
```

ADC (5)

❖ adc.c (3)

```
				else
				{
						int nsamples = nbytes / sizeof(struct adc_msg_s);
						if (nsamples * sizeof(struct adc_msg_s) != nbytes)
						{
								printf("%s : read size %ld is not a multiple of sample size=%d, Ignoring\n", __func__, (long)nbytes, 
sizeof(struct adc_msg_s));
						}
						else
						{
								int i;
								for (i=0; i<nsamples; i++)
								{
										if(sample[i].am_channel == channel)
										{
												data = sample[i].am_data;
												close(fd);
												return data;
										}
								}
						}
				}
		}
}
```

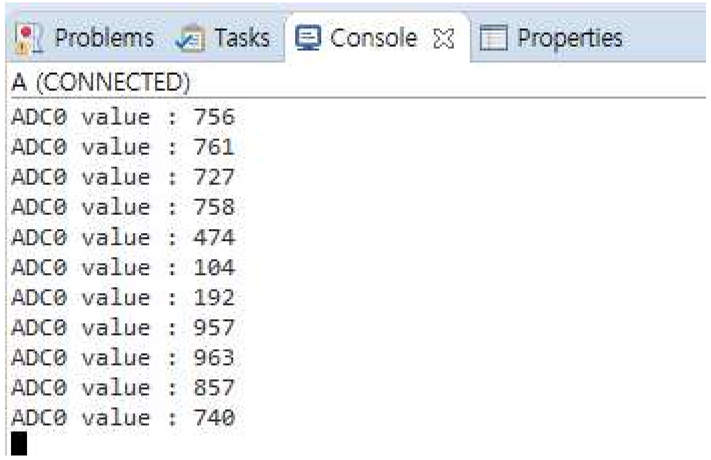
ADC (6)

❖ main_adc.c

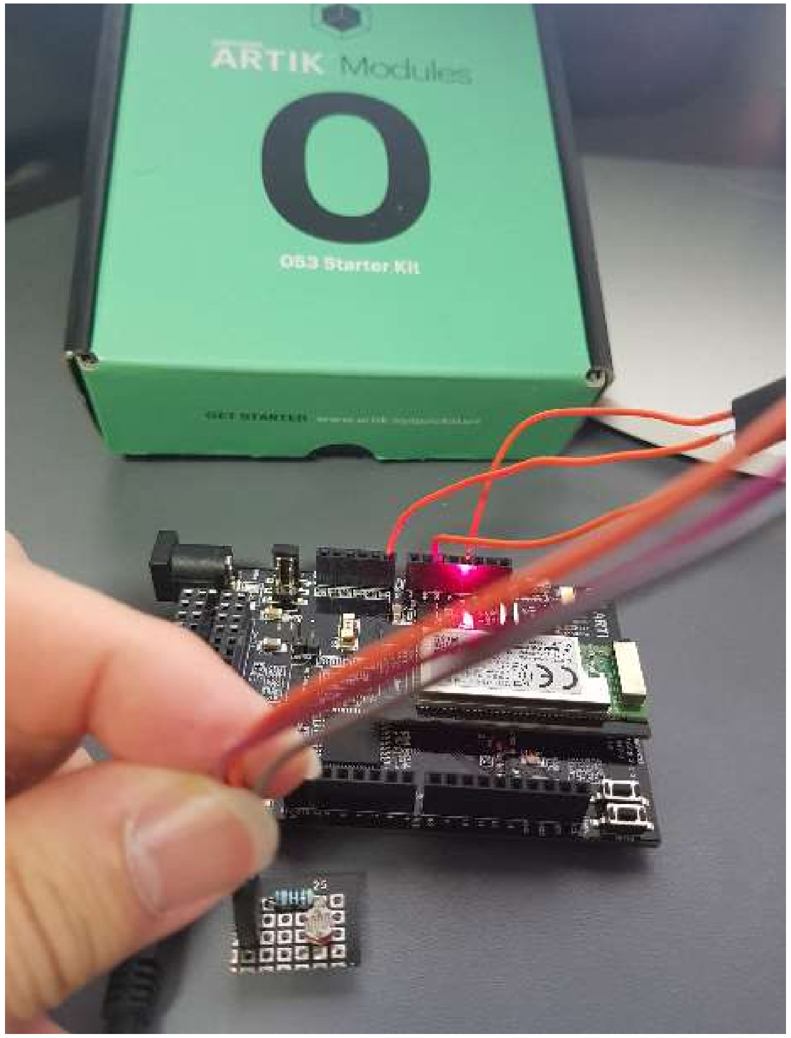
```
#include <stdio.h>
#include "adc.h"

#define ADC_PIN0 0

void main(void)
{
            int32_t val;

            while(1)
            {
                        val = read_adc(ADC_PIN0);
                        printf("ADC%d value : %d \n",ADC_PIN0,val);
                        up_mdelay(1000);
            }
}
```

ADC (7)

❖ **Results**



```
A (CONNECTED)
ADC0 value : 756
ADC0 value : 761
ADC0 value : 727
ADC0 value : 758
ADC0 value : 474
ADC0 value : 104
ADC0 value : 192
ADC0 value : 957
ADC0 value : 963
ADC0 value : 857
ADC0 value : 740
```



CONTROL
ROBOTICS

Thank you