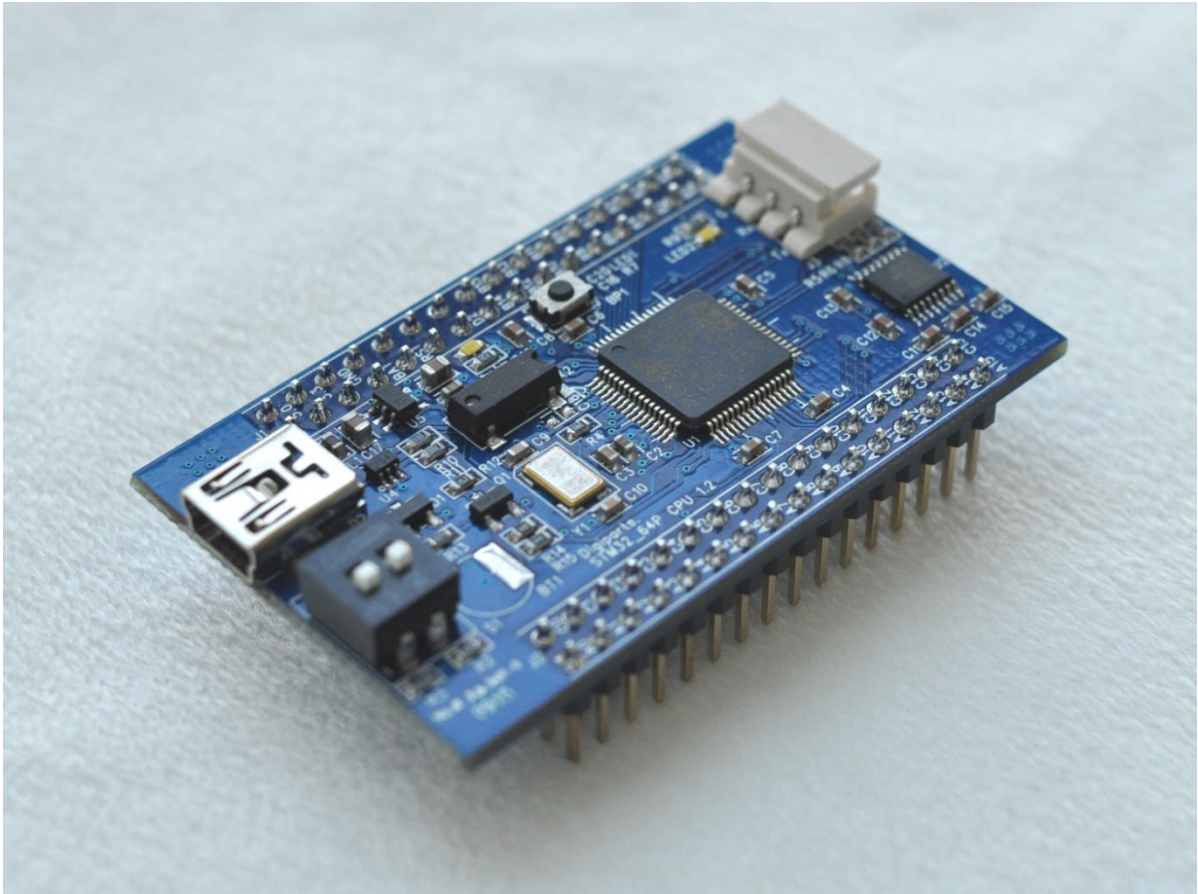


# STM32\_64P MCU BOARD MANUAL



Q/A 지원 : <http://cafe.naver.com/at91sam7s>

보드 관련 자료 : [http://firepooh77.v3webhard.com/Digiparts\\_STM32.zip](http://firepooh77.v3webhard.com/Digiparts_STM32.zip)

온라인 쇼핑몰 : <http://www.devicemart.co.kr/mart7/>

## 목차

1	STM32 64P Overview.....	3
1.1	STM32F103RET6(64P) MCU Features.....	3
1.2	STM32 64P MCU 주요 특징 요약.....	6
1.3	STM32 64P MCU Board 주요 특징 요약.....	7
2	STM32 Key Benefits.....	9
2.1	Leading-edge architecture with Cortex-M3 Core. ....	9
2.2	Outstanding power efficiency.....	10
2.3	High level of integration.....	11
2.4	Motor control .....	12
2.5	Easy development, fast time to market.....	12
3	STM32 64P MCU Board Specifi.....	19
3.1	STM32 64P MCU Board 설명 .....	19
3.2	STM32 64P MCU Board 사양 .....	20
3.3	STM32 64P MCU Board 외부 핀 커넥터.....	21
3.4	STM32 64P MCU Board 사용상 주의 사항 .....	23
4	STM32 ST-Bootloader 사용법 .....	26
4.1	PC Program Install.....	26
4.2	ST-Bootloader 사용법 요약.....	26
4.3	ST-Bootloader 사용법 .....	28
5	WINDE Bootloader 사용법 .....	32
5.1	WINDE Setup .....	32
5.2	Debug Cable 사양.....	35
5.3	Target Board 와 연동 .....	36

## 1 STM32 64P Overview.

### 1.1 STM32F103RET6(64P) MCU Features.

- Core: ARM 32-bit Cortex™-M3 CPU
  - 72 MHz maximum frequency, 1.25 DMIPS/MHz (Dhrystone 2.1) performance at 0 wait state memory access
  - Single-cycle multiplication and hardware Division
- Memories
  - 512 Kbytes of Flash memory
  - 64 Kbytes of SRAM
- Clock, reset and supply management
  - 2.0 to 3.6 V application supply and I/Os
  - POR, PDR, and programmable voltage detector (PVD)
  - 4-to-16 MHz crystal oscillator
  - Internal 8 MHz factory-trimmed RC
  - Internal 40 kHz RC with calibration
  - 32 kHz oscillator for RTC with calibration
- Low power
  - Sleep, Stop and Standby modes
  - VBAT supply for RTC and backup registers
- 2 × 12-bit, 1 μs A/D converters (up to 16 channels)
  - Conversion range: 0 to 3.6 V
  - Triple-sample and hold capability
  - Temperature sensor
- 2 × 12-bit D/A converters
- DMA: 12-channel DMA controller
  - Supported peripherals: timers, ADCs, DAC, SDIO, I2Ss, SPIs, I2Cs and USARTs
- Debug mode
  - Serial wire debug (SWD) & JTAG interfaces
  - Cortex-M3 Embedded Trace Macrocell™
- Up to 51 fast I/O ports
  - 51 I/Os, all mappable on 16 external interrupt vectors, all 5 V-tolerant except for analog inputs
- Up to 11 timers
  - Up to four 16-bit timers, each with up to 4 IC/OC/PWM or pulse counter and quadrature (incremental) encoder input

- 2 × 16-bit motor control PWM timers with dead-time generation and emergency stop
- 2 × watchdog timers (Independent and Window)
- SysTick timer: a 24-bit downcounter
- 2 × 16-bit basic timers to drive the DAC
- Up to 13 communication interfaces
  - Up to 2 × I2C interfaces (SMBus/PMBus)
  - Up to 5 USARTs (ISO 7816 interface, LIN,IrDA capability, modem control)
  - Up to 3 SPIs (18 Mbit/s), 2 with I2S interface multiplexed
  - CAN interface (2.0B Active)
  - USB 2.0 full speed interface
  - SDIO interface
- CRC calculation unit, 96-bit unique ID
- ECOPACK® packages

**Table 2. STM32F103xC, STM32F103xD and STM32F103xE features and peripheral counts**

Peripherals		STM32F103Rx			STM32F103Vx			STM32F103Zx		
Flash memory in Kbytes		256	384	512	256	384	512	256	384	512
SRAM in Kbytes		48	64		48	64		48	64	
FSMC		No			Yes			Yes		
Timers	General-purpose	4								
	Advanced-control	2								
	Basic	2								
Comm	SPI(I <sup>2</sup> S) <sup>(1)</sup>	3(2)								
	I <sup>2</sup> C	2								
	USART	5								
	USB	1								
	CAN	1								
	SDIO	1								
GPIOs		51			80			112		
12-bit ADC		3			3			3		
Number of channels		16			16			21		
12-bit DAC		2			2					
Number of channels		2			2					
CPU frequency		72 MHz								
Operating voltage		2.0 to 3.6 V								
Operating temperatures		Ambient temperatures: -40 to +85 °C / -40 to +105 °C (see Table 10)						Junction temperature: -40 to +125 °C (see Table 10)		
Package		LQFP64			LQFP100 <sup>(2)</sup> , BGA100			LQFP144, BGA144		

1. The SPI2 and SPI3 interfaces give the flexibility to work in an exclusive way in either the SPI mode or the I<sup>2</sup>S audio mode.
2. For the LQFP100 and BGA100 packages, only FSMC Bank1 and Bank2 are available. Bank1 can only support a multiplexed NOR/PSRAM memory using the NE1 Chip Select. Bank2 can only support a 16- or 8-bit NAND Flash memory using the NCE2 Chip Select. The interrupt line cannot be used since Port G is not available in this package.

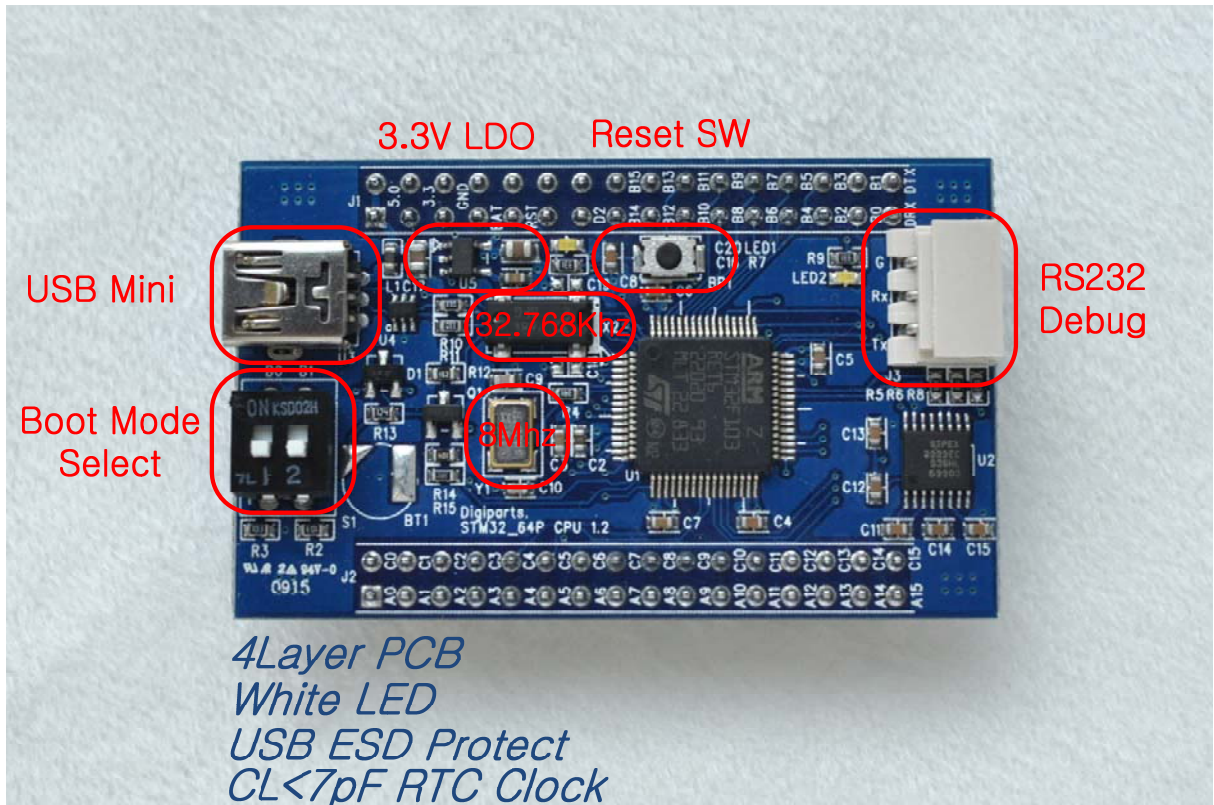
## 1.2 STM32 64P MCU 주요 특징 요약.

- CODE, DATA 버스가 분리된 Harvard Architecture 을 가지고 있다.
- 분기 예측 가능한 Pipeline 구조를 가지고 있다.
- Power Voltage Detect 기능으로 외부 리셋 회로가 필요 없다.
- Clock Security 기능으로 외부 메인 Clock 이상 시 내부 8Mhz RC OSC로 전환 한다. (시스템 안정성 확보)
- 2개의 완전 분리된 Watch-Dog을 지원한다.(시스템 안정성 확보)
- 여러 Mode의 GPIO 셋팅이 가능하다.(전원 투입시 Critical I/O 의 레벨을 안전하게 확보 할 수 있다.)
  - \* Analog Input
  - \* Input Floating
  - \* Input Pull-up
  - \* Input Pull-down
  - \* Output Push-pull
  - \* Output Open-drain
  - \* AF Push-pull
  - \* AF Open-drain
- Tamper Pin 기능은 MCU가 Power Off상태에서도 다음 부팅 시 외부 I/O상태의 바뀜을 알 수 있다.
- 3가지의 Low Power Mode을 제공 하며 Standby Mode에서는 RTC포함하여 3.4uA 전류 소모를 유지한다.
- 72Mhz Full Operation 시에 소모 전류는 33mA 정도 이다.
- 내부 Code용 Flash와 Data용 SRAM이 내장되어 있다.
- 내부 Flash는 10K번 ReWrite가능하다.
- 1개의 USB 2.0 Client Peripheral을 내장하고 있다.
- 1개의 2.0A, 2.0B을 지원하는 CAN Controller가 내장되어 있다.
- USB 와 CAN 은 동시에 사용 할 수 없다.
- 1개의 SDIO Controller(4bit Data bus)가 내장되어 있다.
- 12Bit, 1us Conversion Time을 갖는 3개의 ADC가 내장되어 있다.
- 12Bit, DAC 가 2개 내장되어 있다.
- 51개의 GPIO 가 제공된다.
- 내부 BootLoader가 내장되어 UART1을 통한 프로그램이 가능해, JTAG장비 없이 개발이 가능하다.
- 외부 3.3V Regulator 1개로 동작 가능하다.
- 2.0 ~ 3.6V 범위의 외부 동작 전압 범위를 가진다.
- -40°C ~ 85°C 외부 동작 온도 범위를 가진다.

### 1.3 STM32 64P MCU Board 주요 특징 요약.

- 4Layer PCB 사용함으로써 안정적인 전원공급. Low Noise을 유지함.(내층에서 각각 PWR, GND 할당)
- 4Layer PCB 사용으로 Board의 Size가 소형을 유지함.
- 최적의 Bypass 배치와 라우팅 작업이 되어 있음.
- 외부 2.54mm Pitch 2열 핀 헤더로 접속을 제공함.
- USB 접속 부분에 ESD 소자를 사용하여 접속부의 고장을 미연에 방지함.
- USB Mini 커넥터는 DIP 타입 사용으로 SMD 타입 대비 PAD의 떨어짐을 미연에 방지함.
- USB Slave Pull-up Control FET 장착으로 USB 케이블 탈 부착 없이 USB Slave Enable/Disable을 구현함.
- ST에서 Recommend 하는 CL <= 7pF 이내인 32.768khz RTC장착으로 안정적인 RTC 동작을 보장함.
- SP3232 내장으로 외부 RS232 장비와 바로 연결이 가능함.(외부 핀으로 할당 하여, Baseboard에서도 사용이 가능함)
- Piano SW 장착으로 3가지 BootMode을 모두 지원함.
- Power/Run LED을 고급 White LED로 채용함.
- 기본 ST에서 제공하는 예제를 IAR 컴파일러 용으로 재 가공한 예제를 제공함.
  - 0. App TEMPLATE : STM32\_Template
  - 1. App BOOT : STM32\_BOOT
  - 2. App LED : STM32\_LED
  - 3. App Printf : STM32\_PRINTF
  - 4. App RTC : STM32\_RTC
  - 5. App IWDG : STM32\_IWDG
  - 6. App SysTick : STM32\_SYSTICK
  - 7. App USB\_VCP : STM32\_USB\_VCP
  - 8. App uCOS-II : STM32\_uCOS286





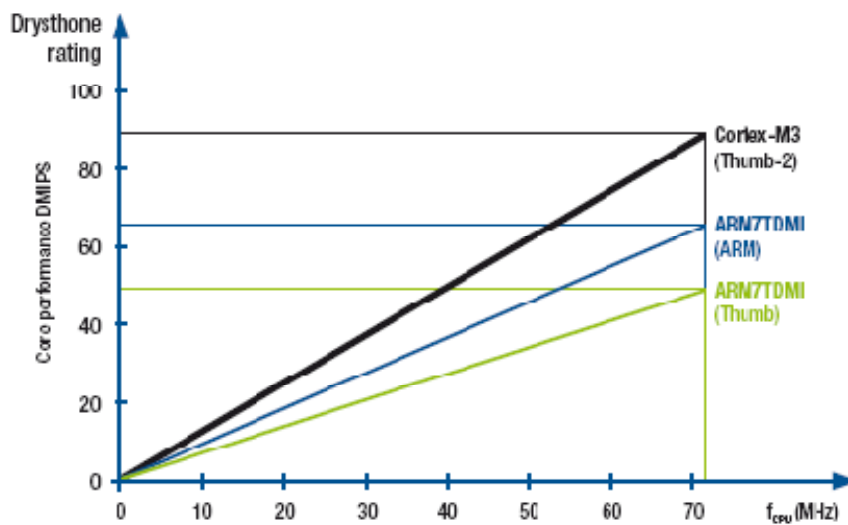


## 2 STM32 Key Benefits.

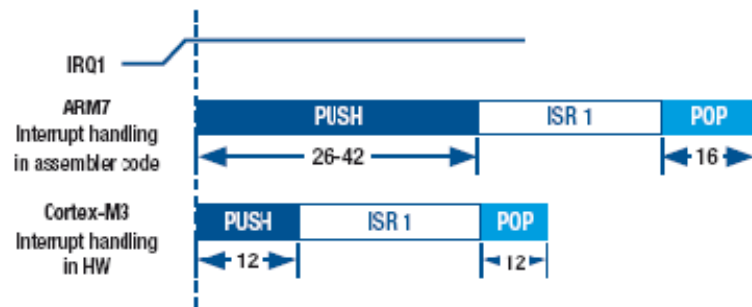
### 2.1 Leading-edge architecture with Cortex-M3 Core.

- Harvard architecture
- 1.25 DMIPS/MHz and 0.19 mW/MHz
- Thumb-2 instruction set brings 32-bit performance with 16-bit code density
- Single cycle multiply and hardware division
- Embedded, fast interrupt controller is now inside the core allowing:
- Excellent real-time behaviour
- Low latency down to six CPU cycles inter-interrupt
- Six CPU cycles wake-up time from low-power mode
- Up to 35% faster and up to 45% less code than ARM7TDMI®

### Cortex-M3 performance versus ARM7TDMI



## Cortex-M3 interrupt versus ARM7TDMI



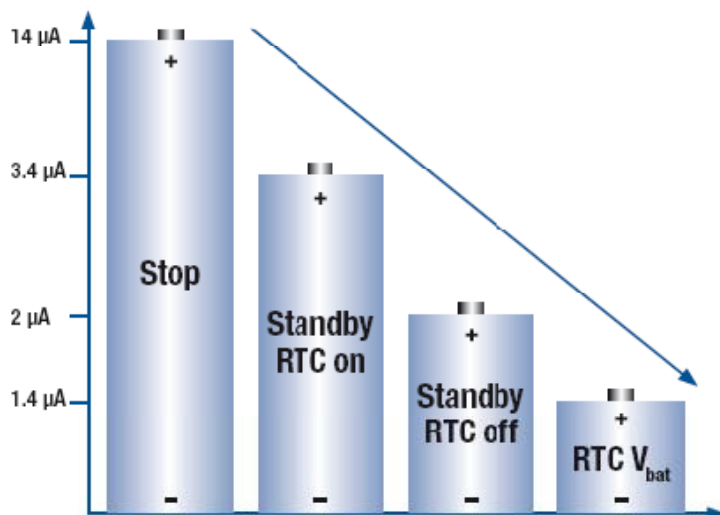
### 2.2 Outstanding power efficiency.

High performance does not mean high power consumption. We have taken special care to address three main energy requirements driven by the market:

- High dynamic power efficiency in running mode
- Extremely low power when the application is in standby
- Low-voltage capability for direct battery operation In run mode, executing from Flash at full 72 MHz CPU speed, the STM32 has a current consumption as low as 27 mA.

### STM32F10x typical current

( $V_{DD}$ : 3.3 V on 128-Kbyte device @ 25 °C)



## 2.3 High level of integration.

- Built-in supervisor reduces need for external components:
  - Power-on reset, low-voltage detect, brown-out detect, watchdog timer with independent clock
- One main crystal drives entire system:
  - Inexpensive 4-16 MHz crystal drives CPU, USB and all peripherals
  - Embedded PLL generates multiple frequencies
  - Optional 32 kHz crystal for RTC
- Embedded factory trimmed 8 MHz RC oscillator can be used as main clock
- Additional low-frequency RC oscillator for RTC or watchdog
- Only 7 external passive components required for base system on LQFP100 package



7 power capacitors only!

## Superior and innovative peripherals

The need for speed	
USB	12 Mbit/s
USART	up to 4.5 Mbit/s
SPI	18 MHz master and slave
I <sup>2</sup> C	400 kHz
GPIO	18 MHz maximum toggle
PWM timer	72 MHz clock input
SDIO	Up to 48 MHz
I2S	From 8 kHz to 48 kHz sampling frequencies
















The need for analog	
ADC	12-bit, 1 $\mu$ s conversion time
DAC	2-channel, 12-bit

## 2.4 Motor control

- The STM32 Performance line embeds features that are perfectly suited to three-phase brushless motor control:
  - Powerful Cortex-M3 core
  - 6 PWM advanced control timers with embedded dead-time generation
  - Numerous PWM outputs allowing multiple DCbrush, stepper or universal motor drives
  - Dual sample and hold ADC, 12-bit resolution, 1  $\mu$ s conversion time
- Free motor control firmware libraries supporting AC induction motor (sensored) and PMSM motor (sensorless, Hall-sensor or encoder) vector control
- Less than 21  $\mu$ s for sensorless vector control loop
- Class B compliancy with the EN/IEC60335-1 norm:
  - Pre-certified full set of self-test routines
- Run your motor in just a few steps:
  - STM3210B-MCKIT full developer kit for vector drives
- For devices starting at 256 Kbytes of Flash, two advanced control PWM timers and three ADCs are on board for dual motor control, triple sample and hold capabilities.

## 2.5 Easy development, fast time to market.

ST 홈페이지에서 각종 F/W LIB 및 여러 Application 에 대한 Source 을 무상으로 제공하여 개발을 용이하게 한다.

Firmware						
Reference	Description	Version	Date	Size	File	File
STM32F10x_FW_Archive	Archive for legacy STM32F10xxx Firmware Library V2.0.3 and all related Firmware packages	2	May-2009			
STM32F10x_StdPeriph_Lib	ARM-based 32-bit MCU STM32F10xxx standard peripheral library	3.0.0	Apr-2009			
STM32_USB-FS-Device_Lib	ARM-based 32-bit MCU STM32F10xxx USB Device Full Speed Library	3.0.1	May-2009			
STM32F10x_CEC_Lib	CEC (consumer electronic control) C library using the STM32F101xx, STM32F102xx and STM32F103xx microcontrollers	2.0.0	May-2009			
STM32F10xFWLib_V2.0.3_Patch1	Patch to fix STM32F10xxx Firmware Library V2.0.3 limitations	1	Apr-2009			
STM3210B-EVAL	STM3210B-EVAL demonstration firmware	2.0.0	May-2009			
STM3210E-EVAL	STM3210E-EVAL demonstration firmware	2.0.0	May-2009			
STM32F10xxx DSP Lib	STM32F10xxx DSP library firmware	2.0.0	May-2009			
STM32F10xxx Speex Lib	STM32F10xxx Speex library firmware STM32, StdPeriph Lib, speex, audio	2.0.0	May-2009			

STM32F10xFWLib 을 웹사이트에서 다운 로드 받으면 다음과 같은 기본적인 F/W 예제를 테스트 할 수 있다.

<b>ADC</b>	<a href="#">3ADCs_DMA</a>	Use 3ADCs in independant continuous conversion mode with ADC1 and ADC3 using DMA transfer from ADC1 and ADC3 to data buffers.
	<a href="#">ADC1_DMA</a>	Use the ADC and DMA to transfer continuously converted data from ADC to a data buffer.
	<a href="#">AnalogWatchdog</a>	Use the ADC analog watchdog to guard continuously an ADC channel.
	<a href="#">ExtLinesTrigger</a>	Trigger ADC regular and injected groups channel conversion using two external line event.
	<a href="#">RegSimul_DualMode</a>	Use ADC1 and ADC2 in regular simultaneous dual mode
<b>BKP</b>	<a href="#">TIMTrigger_AutoInjection</a>	Convert ADC regular group channels continuously using TIM1 external trigger and injected group channels using the auto-injected feature.
	<a href="#">Backup_Data</a>	Store user data in the Backup data registers.
	<a href="#">Tamper</a>	Write/read data to/from Backup data registers and demonstrates the Tamper detection feature.
<b>CAN</b>	<a href="#">LoopBack</a>	Set a communication with the bxCAN in loopback mode.
	<a href="#">Normal</a>	Configure the CAN peripheral to send and receive CAN frames in normal mode.
<b>CortexM3</b>	<a href="#">BitBand</a>	Use CortexM3 Bit-Band access to perform atomic read-modify-write and read operations on a variable in SRAM.
	<a href="#">Mode_Privilege</a>	Modify CortexM3 Thread mode privilege access and stack.
<b>CRC</b>	<a href="#">Example</a>	Use the CRC calculation unit to get a CRC code of a given buffer of data word(32-bit).
<b>DAC</b>	<a href="#">DualModeDMA_SineWave</a>	Use DAC channels in dual mode with DMA to generate the same sine wave signal in 12bit right data alignment mode.
	<a href="#">OneChannel_NoiseWave</a>	Use DAC channel1 to generate a fixed signal with noise wave in 12bit left data alignment mode.

	<a href="#">OneChannelDMA_Escalator</a>	Use DAC channel1 with DMA to generate an escalator signal in 8bit right alignment mode
	<a href="#">TwoChannels_TriangleWave</a>	Use DAC channels separately to generate two signals with different triangle waves in 12bit right data alignment mode.
<b>DMA</b>	<a href="#">ADC_TIM1</a>	Use a DMA channel to transfer continuously a data from a peripheral (ADC) to another (TIM1) supporting DMA transfer.
	<a href="#">FLASH_RAM</a>	Use a DMA channel to transfer a word data buffer from memory (Flash) to memory (RAM).
	<a href="#">FSMC</a>	Use two DMA1 channels to transfer a word data buffer from embedded FLASH to external memory through FSMC and from this external memory to embedded RAM.
	<a href="#">I2C_RAM</a>	Use two DMA channels to transfer a data buffer from memory to I2C2 through I2C1.
	<a href="#">SPI_RAM</a>	Use four DMA channels to transfer a data buffer from memory to SPI2 through SPI1 and a second data buffer from memory to SPI1 through SPI2 in full-duplex mode.
<b>EXTI</b>	<a href="#">Example</a>	Configure an external interrupt line.
<b>FLASH</b>	<a href="#">Program</a>	Program the STM32F10x FLASH.
	<a href="#">Write_Protection</a>	Enable and disable the write protection for the STM32F10x FLASH.
<b>FSMC</b>	<a href="#">NAND</a>	Use the FSMC firmware library and an associate driver to communicate with a 8-Bit NAND memory: NAND512W3A2. Write into all memory, Read and verify the contents.
	<a href="#">NOR</a>	Use the FSMC firmware library and an associate driver to communicate with a 16-Bit NOR memory: M29W128FL, M29W128GL or S29GL128P. Write into all memory, Read and verify the contents .
	<a href="#">NOR_CodeExecute</a>	Build an application to be loaded into the NOR memory mounted on STM3210E-EVAL board then execute it from internal Flash.
	<a href="#">SRAM</a>	Use the FSMC firmware library and an associate driver to communicate with a 16-Bit SRAM memory: IS61WV51216BLL. Write into all memory, Read and verify the contents.
	<a href="#">SRAM_DataMemory</a>	Use the external SRAM mounted on STM3210E-EVAL board as program data memory and internal SRAM for Stack.

GPIO	<a href="#">IOToggle</a>	Use the GPIO BSRR (Port bit set/reset register) and BRR (Port bit reset register) for IO toggling. These registers allow modifying only one or several GPIO pins in a single atomic write access.
	<a href="#">JTAG_Remap</a>	Use the JTAG IOs as standard GPIOs and gives a configuration sequence.
I2C	<a href="#">10bitAddress</a>	Transfer a data buffer from I2C1 to I2C2 in 10-bit addressing mode.
	<a href="#">DualAddress</a>	Transfer two data buffer from I2C1 to I2C2 through its two addresses in the same application.
	<a href="#">Interrupt</a>	Transfer a data buffer from master transmitter (I2C1) to slave receiver (I2C2) and from slave transmitter (I2C2) to master receiver (I2C1) using interrupts.
	<a href="#">M24C08_EEPROM</a>	Use the I2C firmware library and an associate I2C EEPROM driver to communicate with an M24C08 EEPROM.
	<a href="#">SMBus</a>	Send an ARP command from I2C1 to I2C2 in SMBus mode.
I2S	<a href="#">Interrupt</a>	Describe how to configure and use the I2S mode interrupts with 16bits and 24 bits data in 32 bits packet frames.
	<a href="#">SPI_I2S_Switch</a>	Describe how to configure and use the I2S mode in alternation with the SPI mode. Simple communication is performed between two SPIs then between two I2Ss (raw data transfer and verification).
IWDG	<a href="#">Example</a>	Reload at regulate period the IWDG counter using the SysTick interrupt.
LIB_DEBUG	<a href="#">Example</a>	Demonstrates how to declare a dynamic peripherals pointer used for Debug mode. When the "USE_FULL_ASSERT" label (stm32f10x_conf.h file) is uncommented, the assert macro is expanded and run time checking is enabled in the standard peripherals library drivers code. The run-time checking allows checking that all the library functions input value lies within the parameter allowed values.
NVIC	<a href="#">DMA_WFIMode</a>	Enter the system to WFI mode with DMA transfer enabled and wake-up from this mode by the DMA End of Transfer interrupt.
	<a href="#">IRQ_Channels</a>	Use of the Nested Vectored Interrupt Controller (NVIC) and IRQ Channels configuration.
	<a href="#">Priority</a>	Use of the Nested Vectored Interrupt Controller (NVIC) and priority mechanism



(PreemptionPriority , SubPriority).

Use the NVIC firmware library to set the CortexM3 vector table in a specific address other than default. This can be used to build program which will be loaded into Flash memory by an application previously programmed in the sector0 of the Flash memory. Such application can be In-Application Programming (IAP, through USART) or Device Firmware Upgrade (DFU, through USB).

#### [VectorTable\\_Relocation](#)

### PWR

#### [STANDBY](#)

Enter the system to STANDBY mode and wake-up from this mode using: external RESET, RTC Alarm or WKUP pin.

#### [STOP](#)

Enter the system to STOP mode and wake-up using EXTI Line interrupts. The EXTI Line sources are Key Button and RTC Alarm.

### RCC

#### [Example](#)

Configure the system clock source, AHB, APB2 and APB1 prescaler. It demonstrates also the Clock Security System (CSS) which handles the High Speed External clock (HSE) failure detection and system clock back-up.

### RTC

#### [Calendar](#)

Explain how to use the RTC peripheral. As an application example, it demonstrates how to setup the RTC peripheral, in terms of prescaler and interrupts, to be used to keep time and to generate Second interrupt.

#### [LSI Calib](#)

Describe how to use the LSI calibration to get RTC accurate time base.

### SDIO

#### [Example](#)

Use the SDIO firmware library and an associate driver to perform read/write operations on the SD Card memory.

### SPI

#### [CRC](#)

Set a communication between two SPIs in full-duplex mode and performs a transfer from Master to Slave and Slave to Master followed by CRC transmission.

#### [DMA](#)

Set a communication between the two SPIs in simplex mode and performs a transfer from Master in polling mode to the Slave in DMA receive mode. The NSS pin is managed by hardware.

#### [FullDuplex\\_SoftNSS](#)

Set a communication between the two SPIs in full-duplex mode and performs a transfer from Master to Slave and then Slave to Master in the same application with software NSS management.

#### [M25P64\\_FLASH](#)

Use the SPI firmware library and an associated SPI FLASH driver to communicate with an M25P64 FLASH.

#### [Simplex Interrupt](#)

Set a communication between two SPIs in simplex mode and performs a data buffer

transfer from Master to Slave using TxE interrupt for master and RxNE interrupt for slave.

<b>SysTick</b>	<a href="#">Example</a>	Configure the SysTick to generate a time base equal to 1ms.
<b>TIM</b>	<a href="#">6Steps</a>	Configure the TIM1 peripheral to generate 6 Steps.
	<a href="#">7PWM_Output</a>	Configure the TIM1 peripheral to generate 7 PWM signals with 4 different duty cycles.
	<a href="#">Cascade_Synchro</a>	Synchronize TIM peripherals in cascade mode.
	<a href="#">ComplementarySignals</a>	Configure the TIM1 peripheral to generate three complementary TIM1 signals, to insert a defined dead time value, to use the break feature and to lock the desired parameters.
	<a href="#">DMA</a>	Use DMA with TIM1 Update request to transfer Data from memory to TIM1 Capture Compare Register3.
	<a href="#">ExtTrigger_Synchro</a>	Synchronize TIM peripherals in cascade mode with an external trigger.
	<a href="#">OCActive</a>	Configure the TIM peripheral to generate four different signals with four different delays.
	<a href="#">OCInactive</a>	Configure the TIM peripheral in Output Compare Inactive mode with the corresponding Interrupt requests for each channel.
	<a href="#">OCToggle</a>	Configure the TIM peripheral to generate four different signals with four different frequencies.
	<a href="#">OnePulse</a>	Use the TIM peripheral to generate One pulse after a Rising edge of an external signal is received in Timer Input pin.
	<a href="#">Parallel_Synchro</a>	Synchronize TIM peripherals in parallel mode.
	<a href="#">PWM_Input</a>	Use the TIM peripheral to measure the frequency and duty cycle of an external signal.
	<a href="#">PWM_Output</a>	Configure the TIM peripheral in PWM (Pulse Width Modulation) mode.
	<a href="#">TIM1_Synchro</a>	Synchronize TIM1 and Timers (TIM3 and TIM4) in parallel mode.
	<a href="#">TimeBase</a>	Configure the TIM peripheral in Output Compare Timing mode with the corresponding Interrupt requests for each channel in order to generate 4 different time bases.
<b>USART</b>	<a href="#">DMA Interrupt</a>	Provide a basic communication between USART1 and USART2 using DMA capability, flags and interrupts.
	<a href="#">DMA Polling</a>	Provide a basic communication between USART1 and USART2 using DMA capability.

<a href="#">HalfDuplex</a>	Provide a basic communication between USART1 and USART2 in Half-Duplex mode using flags.
<a href="#">HyperTerminal_HwFlowControl</a>	Use the USART with hardware flow control and communicate with the Hyperterminal.
<a href="#">HyperTerminal_Interrupt</a>	Use the USART1 interrupts to communicate with the hyperterminal.
<a href="#">Interrupt</a>	Provide a basic communication between USART1 and USART2 using interrupts.
<a href="#">IrDA Receive</a>	Use IrDA receiver mode.
<a href="#">IrDA Transmit</a>	Use IrDA transmitter mode.
<a href="#">MultiProcessor</a>	Use the USART in multi-processor mode.
<a href="#">Polling</a>	Provide a basic communication between USART1 and USART2 using flags.
<a href="#">Printf</a>	Retarget the C library printf function to the USART.
<a href="#">Smartcard</a>	Use the USART in Smart Card mode.
<a href="#">Synchronous</a>	Provide a basic communication between USART1 (Synchronous mode) and SPI1 using flags.
<b>WWDG</b>	<a href="#">Example</a> Update at regulate period the WWDG counter using the Early Wakeup interrupt (EWI).

### 3 STM32 64P MCU Board Spec.

#### 3.1 STM32 64P MCU Board 설명

- 4Layer PCB 사용으로 안정적 전원으로 Low Noise을 유지함.(내층에 각각 PWR, GND 할당)
- 4Layer PCB 사용으로 Board의 Size가 소형을 유지함.
- 최단, 최적의 Bypass 배치와 라우팅 작업이 되어 있음.
- 외부 2.54mm Pitch 2열 핀 헤더로 접속을 제공함.
- USB 접속 부분에 ESD 소자를 사용하여 접속부의 고장을 미연에 방지함.
- USB Mini 컨넥터는 DIP 타입 사용으로 SMD 타입 대비 PAD의 떨어짐을 미연에 방지함.
- USB Slave Pull-up Control FET 장착으로 USB 케이블 탈 부착 없이 USB Slave Enable/Disable을 구현함.
- ST에서 Recomend하는  $CL \leq 7pF$  이내인 32.768khz RTC장착으로 안정적인 RTC 동작을 보장함.
- SP3232 내장으로 외부 RS232 장비와 바로 연결이 가능함.(외부핀으로 할당 하여, Baseboard에서도 사용이 가능함)
- Piano SW 장착으로 3가지 BootMode을 모두 지원함.
- Power/Run LED을 고급 White LED로 채용함.

NAVER CAFE 에서 각종 Q/A 지원.

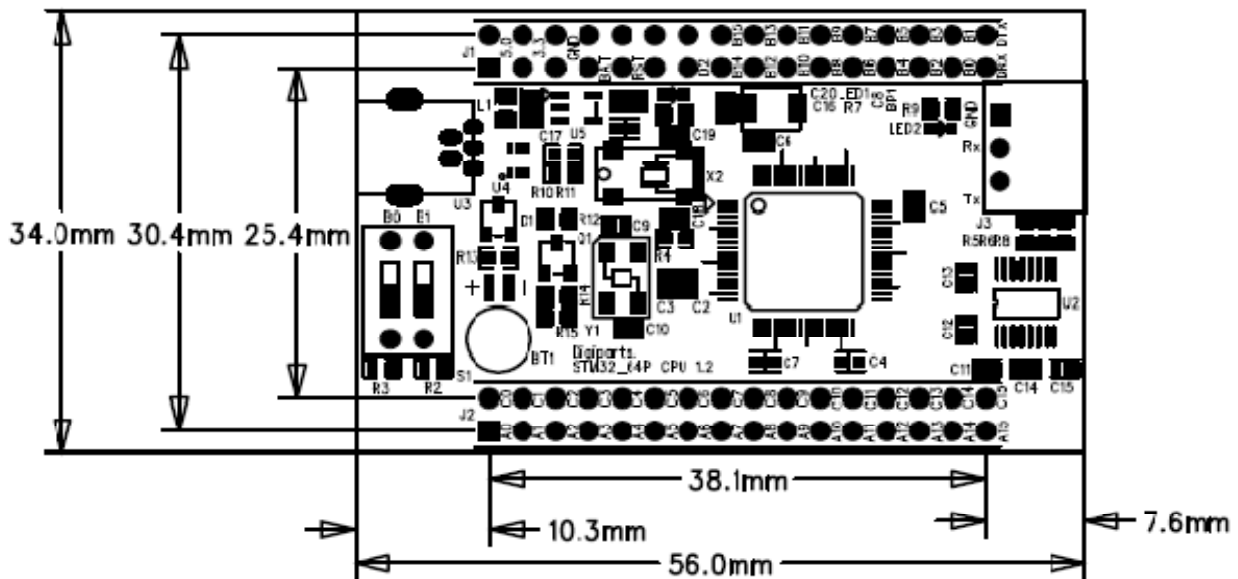
<http://cafe.naver.com/at91sam7s.cafe>

ST 홈페이지

<http://www.st.com>

### 3.2 STM32 64P MCU Board 사양

- CPU : STM32F103RET6
- PCB : 4Layer 1.6T (내층 PWR,GND 할당)
- POWER : 3.3V LOW Noise LDO (300mA)
- Main Clock : Mini SMD 8MHz Crystal (CPU 동작 속도-PLL 동작 72Mhz)
- RTC Clock : Mini SMD 32.768Khz
- LED : 전원 확인 1 개, 응용 프로그램용 1 개
- USB Mini : Board 전원 공급용 및 USB Slave 통신용. (ESD 소자 적용)
- DEBUG : RS232 용 3 핀 Debug Port 1 개
- 외부 핀형태 : 2 개의 25x2pin DIP 타입 (2.54mm 간격)
- 크기 : 34mm \* 56mm
- Switch : Reset SW 1ea, Boot 선택용 Piano SW 1ea



## 3.3 STM32 64P MCU Board 외부 핀 커넥터

번호	J1	번호	J1	번호	J2	번호	J2
1	+5V(I/O)	2	+5V(I/O)	1	PA00	2	PC00
3	+3.3V(O)	4	+3.3V(O)	3	PA01	4	PC01
5	GND	6	GND	5	PA02	6	PC02
7	Vbat	8	NC	7	PA03	8	PC03
9	nRST	10	NC	9	PA04	10	PC04
11	NC	12	NC	11	PA05	12	PC05
13	PD02	14	NC	13	PA06	14	PC06
15	PB14	16	PB15	15	PA07	16	PC07
17	PB12	18	PB13	17	PA08	18	PC08
19	PB10	20	PB11	19	PA09	20	PC09
21	PB08	22	PB09	21	PA10	22	PC10
23	PB06	24	PB07	23	PA11	24	PC11
25	PB04	26	PB05	25	PA12	26	PC12
27	PB02	28	PB03	27	PA13	28	PC13
29	PB00	30	PB01	29	PA14	30	PC14
31	RS232-RX	32	RS232-TX	31	PA15	32	PC15

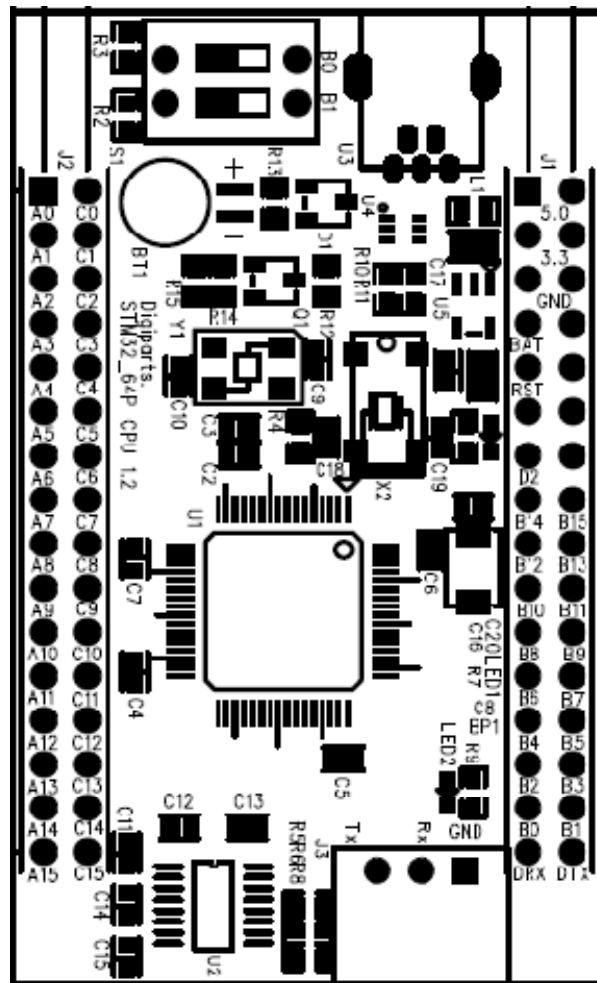


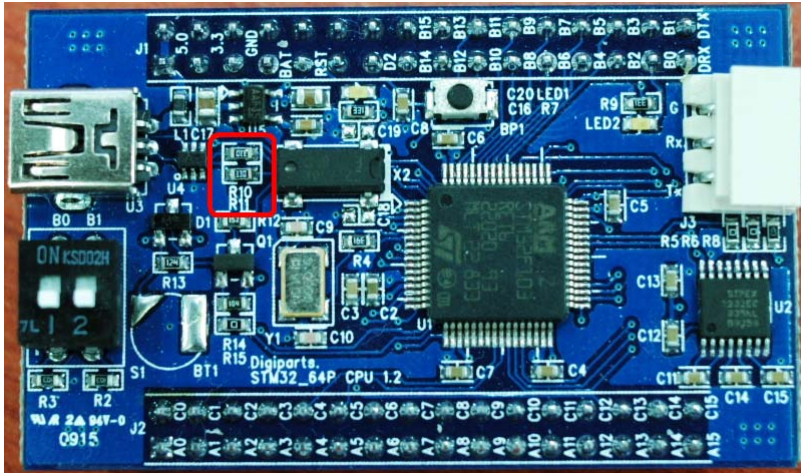
그림 2 컨넥터 배치



### 3.4 STM32 64P MCU Board 사용상 주의 사항

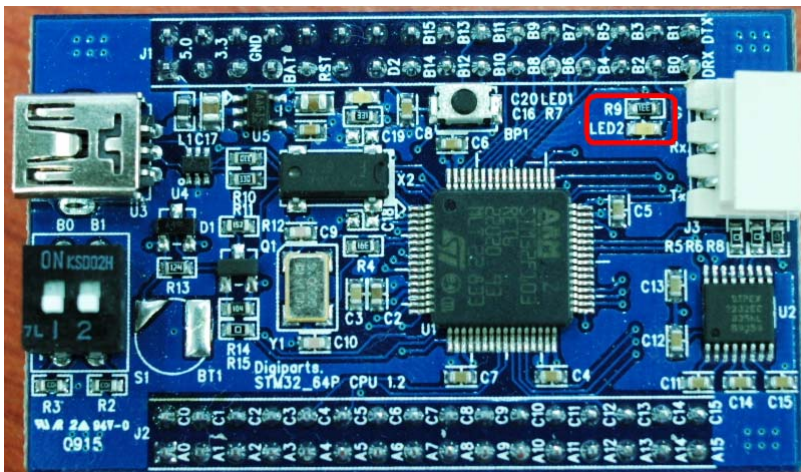
#### 3.4.1 주의 사항 1

PA11, PA12 을 USB 로 사용하지 않고 범용 I/O 로 사용하기 위해서는 MCU Board 의 R10, R11 을 제거하고 사용함.



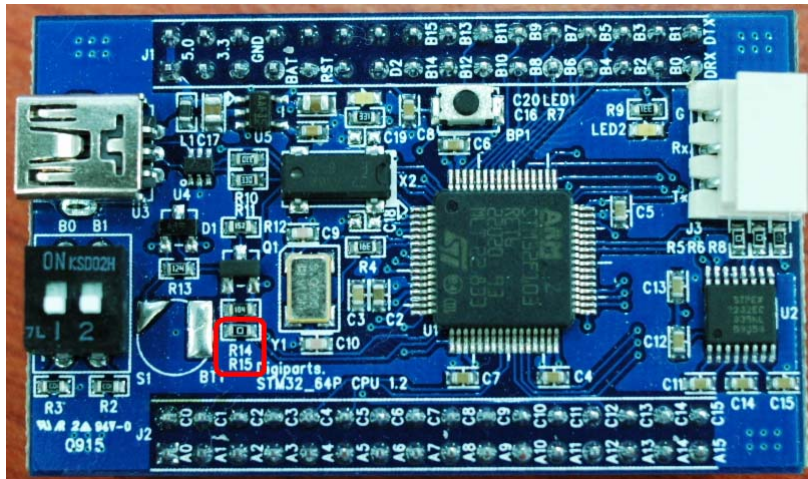
#### 3.4.2 주의 사항 2

PB00 을 MCU Board 에서의 LED 제어용으로 사용하지 않고, 범용 I/O 로 사용하기 위해서는 LED2 을 제거 하고 사용함.



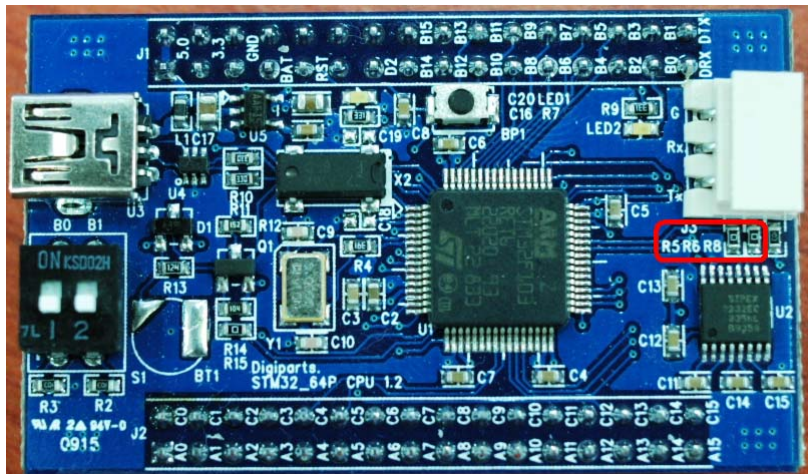
### 3.4.3 주의 사항 3

PB01 을 USB Enable 신호로 사용하지 않고, 범용 I/O 로 사용하기 위해서는 R15 을 제거 하고 사용함.



### 3.4.4 주의 사항 4

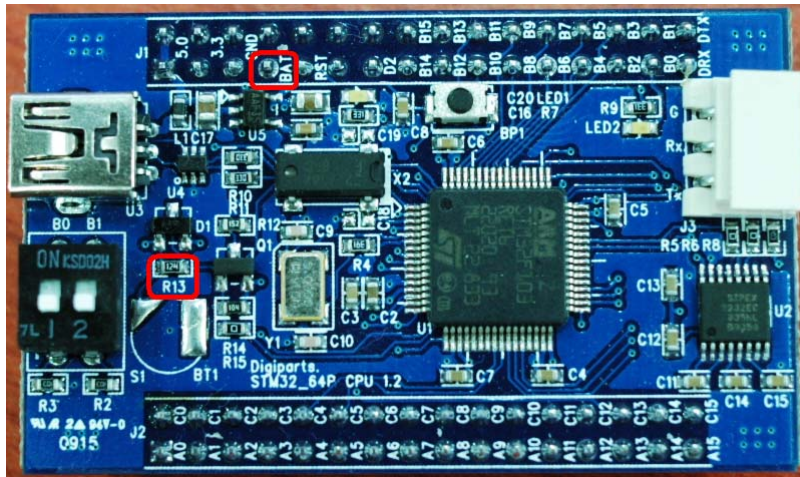
PA09, PA10 을 SP3232 을 이용한 Serial Debug Port 로 사용하지 않고, 범용 I/O 로 사용하기 위해서는 R5, R6 을 제거 하고 사용함.





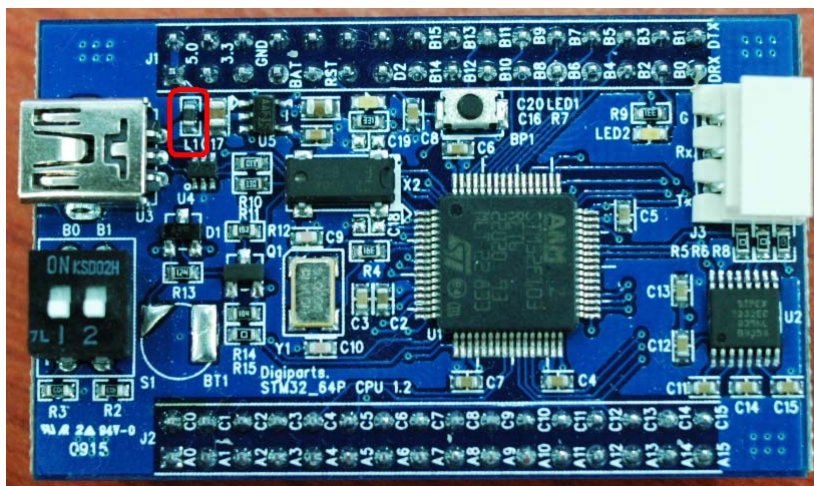
### 3.4.5 주의 사항 5

외부 Backup 전원을 사용하기 위해서는 R13 을 제거하고, J1 의 7 번 핀으로 Backup 전원을 공급함.



### 3.4.6 주의 사항 6

USB 파워를 사용하지 않고, 외부 전원 사용시 BEAD L1 을 제거 하고 사용함.



## 4 STM32 ST-Bootloader 사용법

### 4.1 PC Program Install

Flash loader demonstrator (v1.3)는 ST사에서 제공하는 Window 프로그램으로써, ST에서 만들어 내고 있는 STM 계열의 내부 Bootloader와 연동하여 MCU 내부의 Flash write 기능을 제공하는 프로그램입니다. 따라서 JTAG 장비 없이, MCU 내부 Flash에 바로 Read/Write 할 수 있습니다.

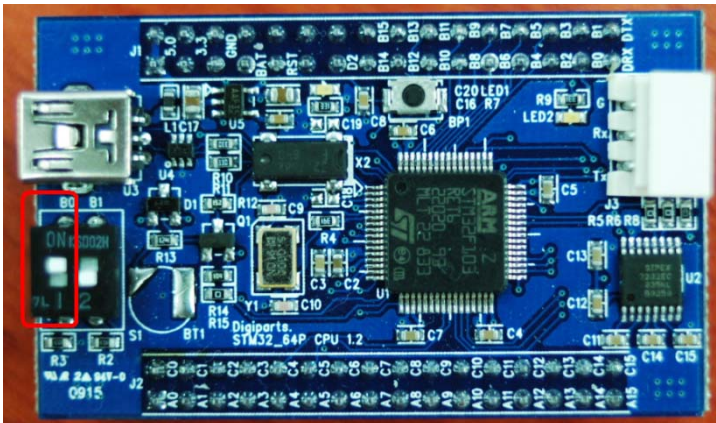
STM32는 내부에 BootLoader가 기본적으로 내장되어 있으며 외부 Boot0/1 핀을 통해 선택적으로 동작하게 됩니다. 현재 Ver1.3가 Release되어 있으며 ST 홈페이지에서 다운 받아 볼 수 있습니다.

PC용 프로그램은 다음과 같이 설치하면 됩니다.

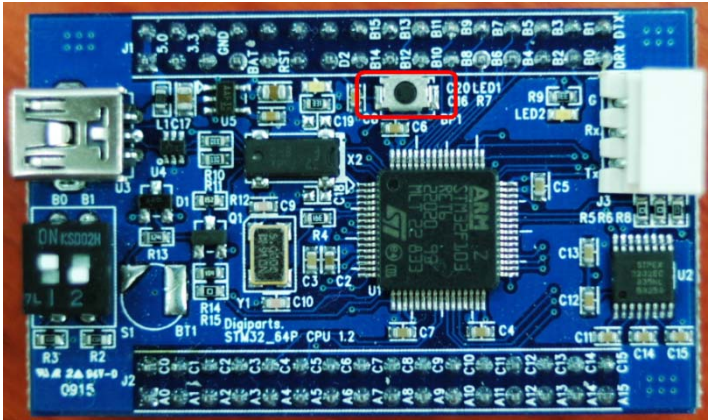
현재 ST에서는 Windows 98, Millennium, 2000, XP or Vista 운영체제를 지원하고 있다고 합니다.

### 4.2 ST-Bootloader 사용법 요약

- ① MCU Board의 B0을 ON 시켜준다.(USB Cable로 파워를 공급한다.)



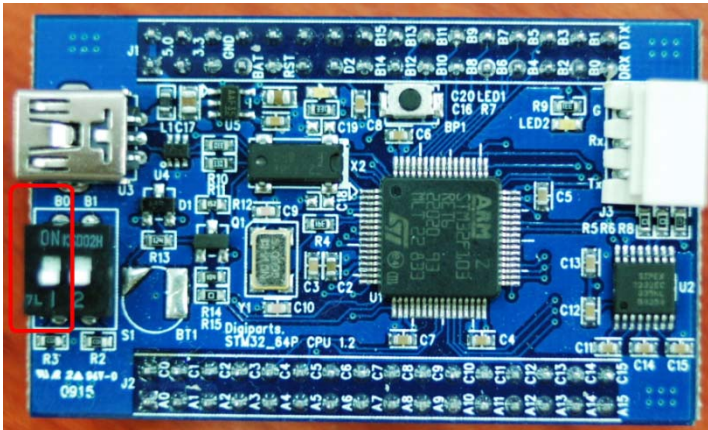
- ② MCU Board 의 Reset Button 을 한번 눌러 Reset 을 시켜줌.



- ③ PC 프로그램인 Flash loader demonstrator 을 실행 시킨다.

- ④ Flash 에 Write 하거나, Read 동작을 수행 한다.

- ⑤ MCU Board 의 B0 을 다시 OFF 시켜준다.



- ⑥ MCU Board 의 Reset Button 을 한번 눌러 Reset 을 시켜주면 Flash 의 내용이 실행된다.

### 4.3 ST-Bootloader 사용법

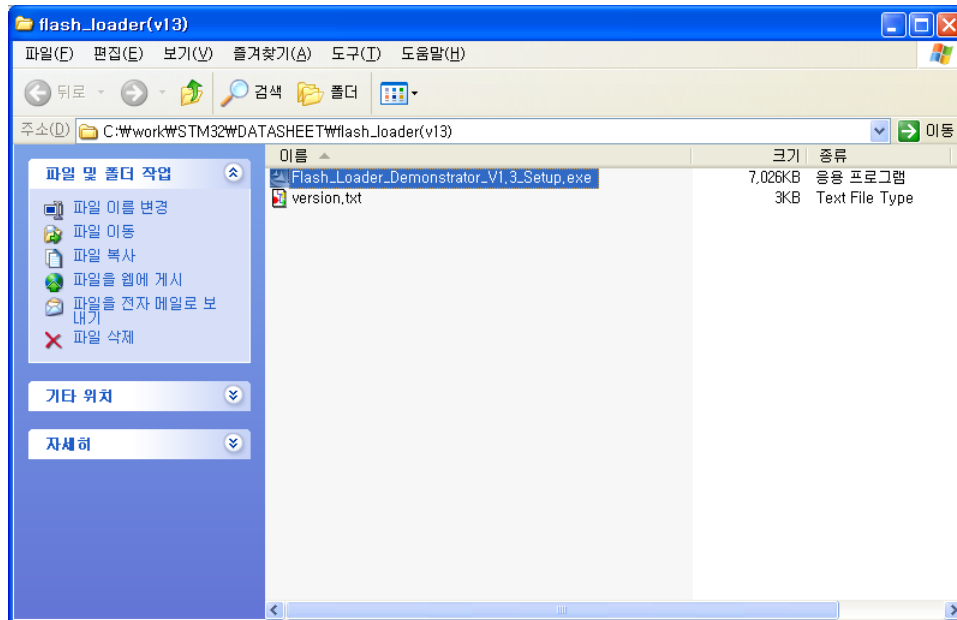


그림 3 Flash\_Loader\_Demonstrator\_V1.3\_Setup.exe 을 실행.



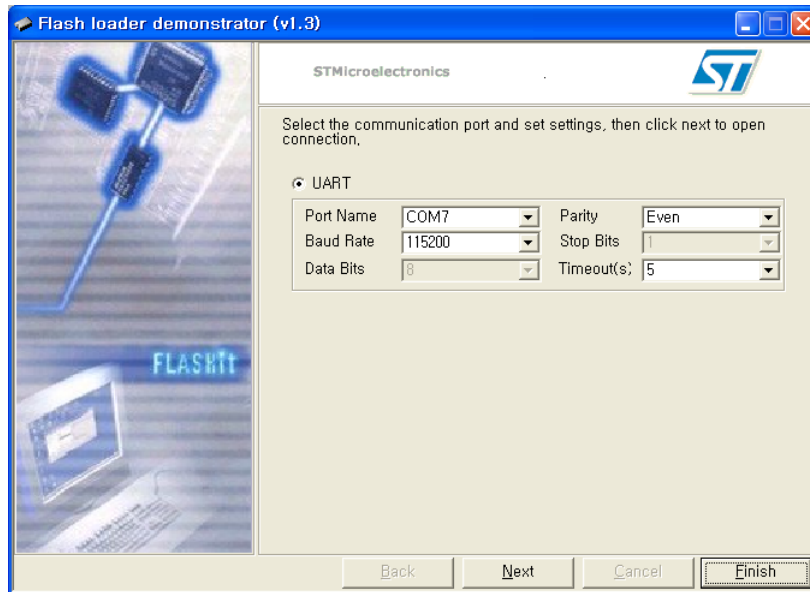


그림 4 프로그램 Install 후 실행 프로그램을 실행 하면 위와 같은 화면이 나옴.

MCU Board의 B0 핀을 On 시키고 Reset 이나 Power Reset 을 시켜줌. 그리고 Port Name 만 실제 MCU Board 와 연결된 Port 와 일치시켜주고, 나머지 값들은 Default 상태로 남겨둠.

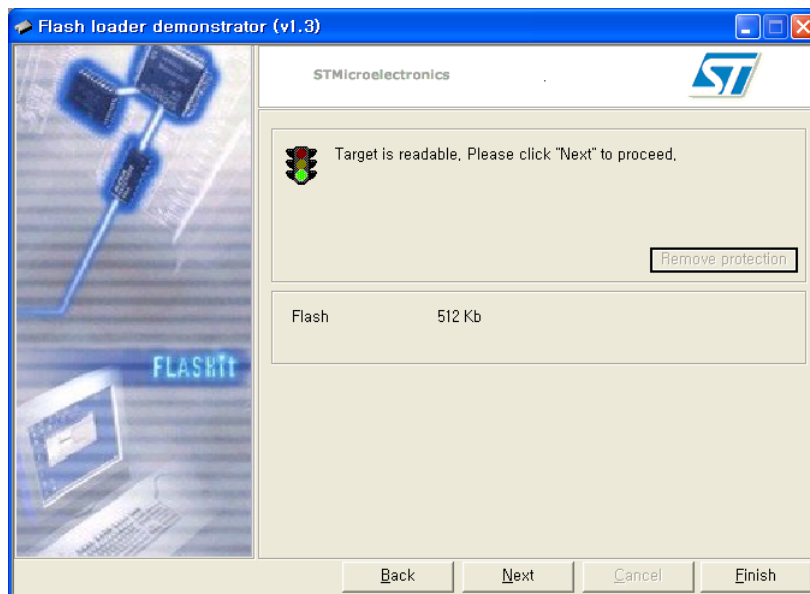


그림 5 Target 과 정상적인 통신이 이루어지면 나타나는 화면.



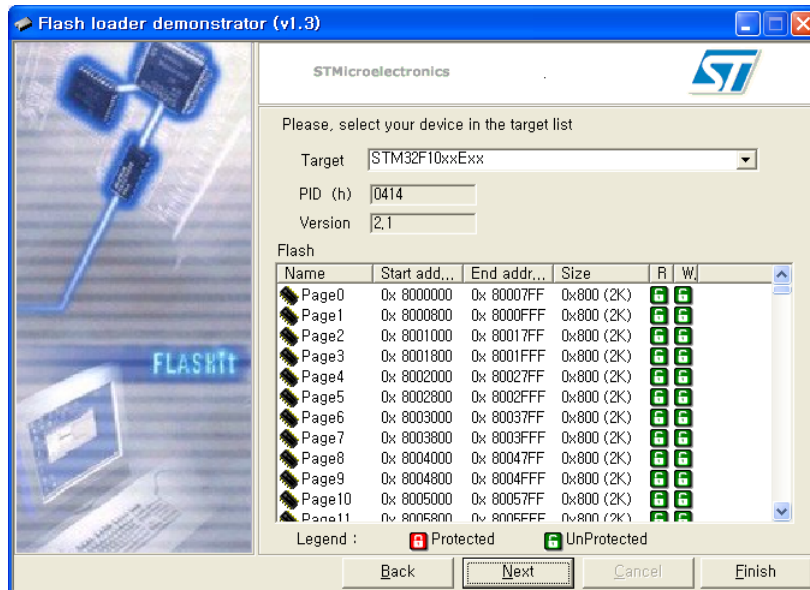


그림 6 내부 Flash 상태를 보여주는 화면.

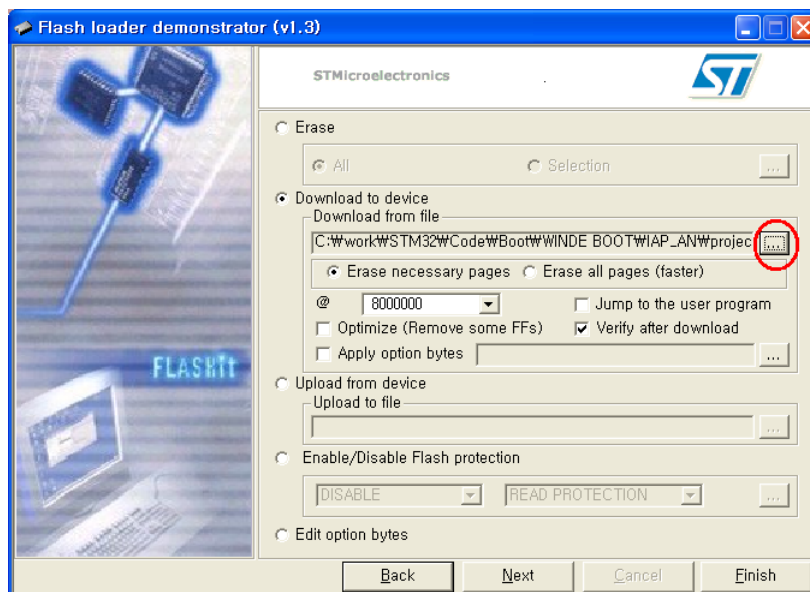


그림 7 Flash 에 Write 하게될 Binary 을 선택하고 Next 선택함.

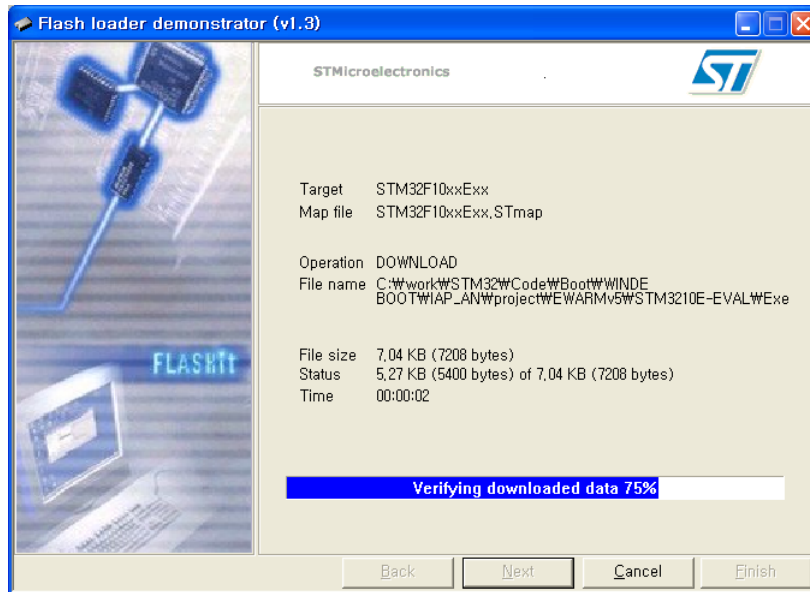


그림 8 Flash 에 Write 하고 Verify 하는 화면.

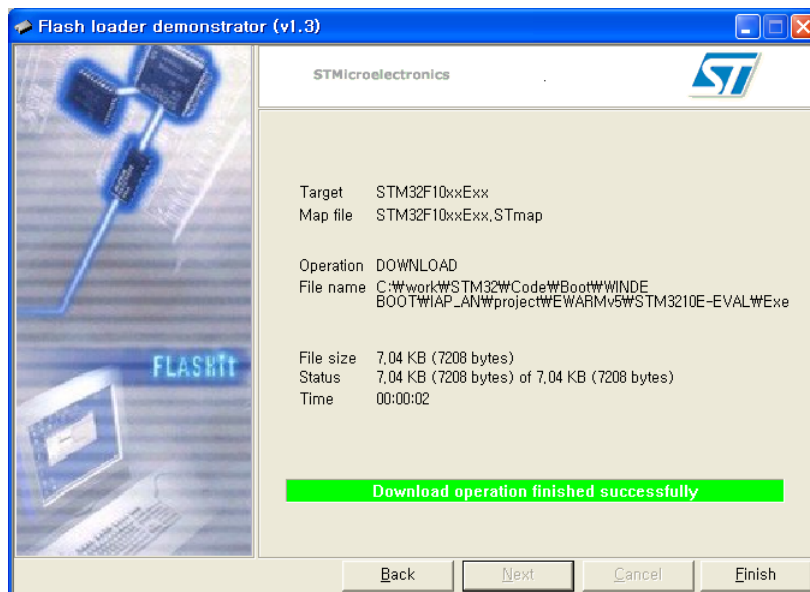


그림 9 Flash Write/Verify 가 완료된 화면.

## 5 WINDE Bootloader 사용법

### 5.1 WINDE Setup

WINDE 프로그램은 일반 UART 통신 터미널 프로그램입니다.

F/W Download 기능을 추가로 가지고 있어, UART 로 Debug 을 하면서 F/W Update 을 할 수 있는 편리한 프로그램 입니다.

WINDE 와의 통신이 되려면, 먼저 STM32 MCU Board 에 WINDE 용 Bootloader Binary 을 Flash loader demonstrator (v1.3) 프로그램을 이용하여 Write 해야 합니다.

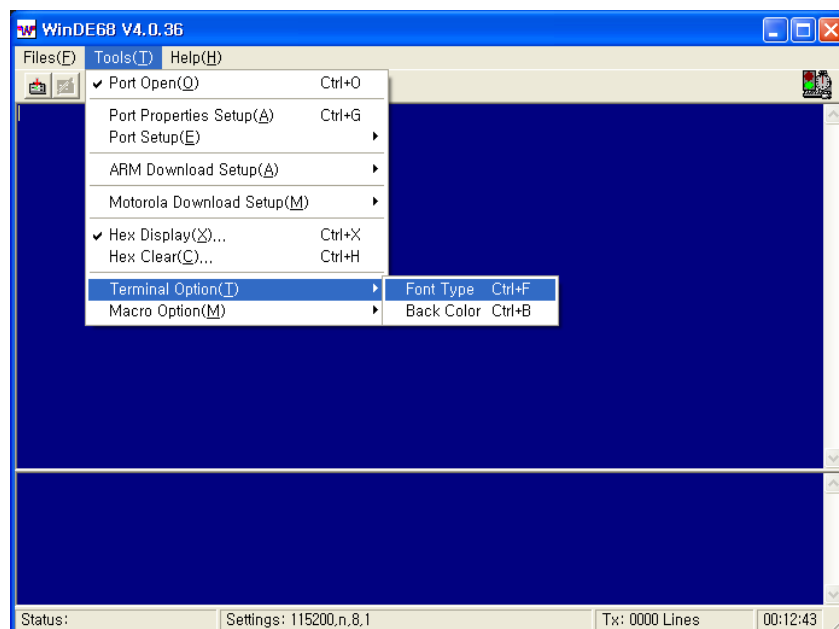


그림 10 Font 을 자신의 PC 환경에 맞게 설정함.(굴리체, 10Size 권장)

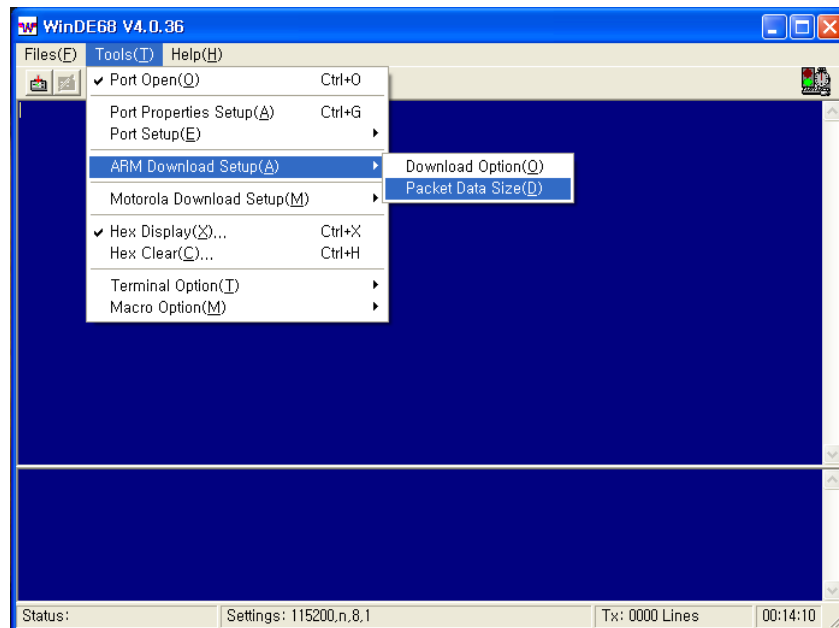


그림 11 Packet Data Size 을 설정 하는 메뉴로 들어감.

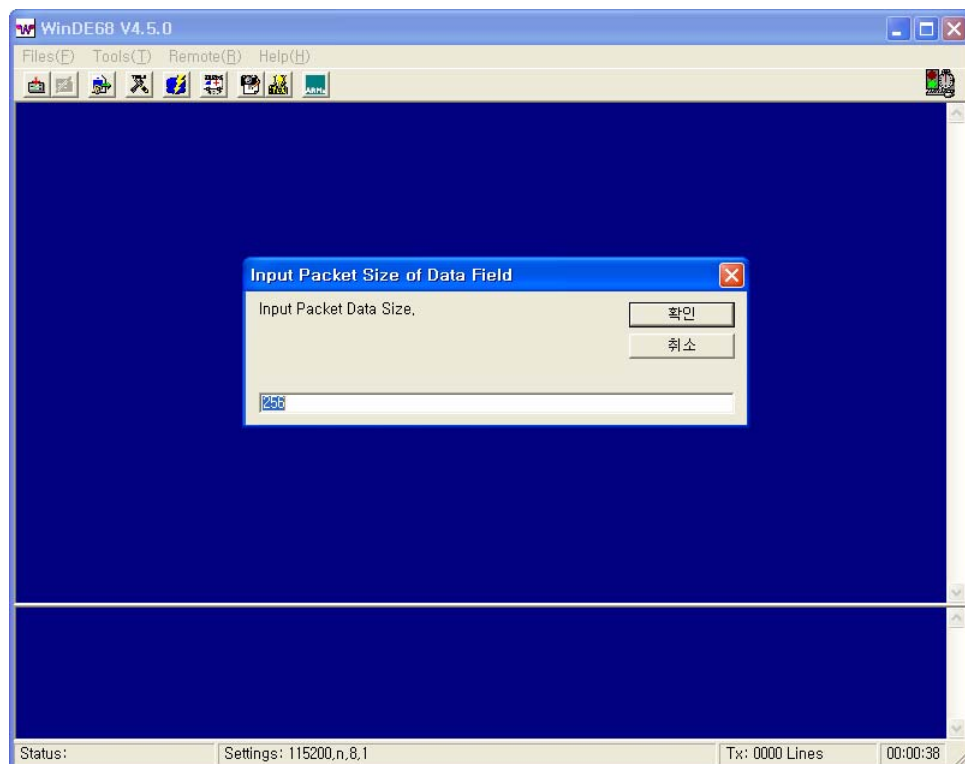


그림 12 Packet Data Size 을 256 으로 설정함.

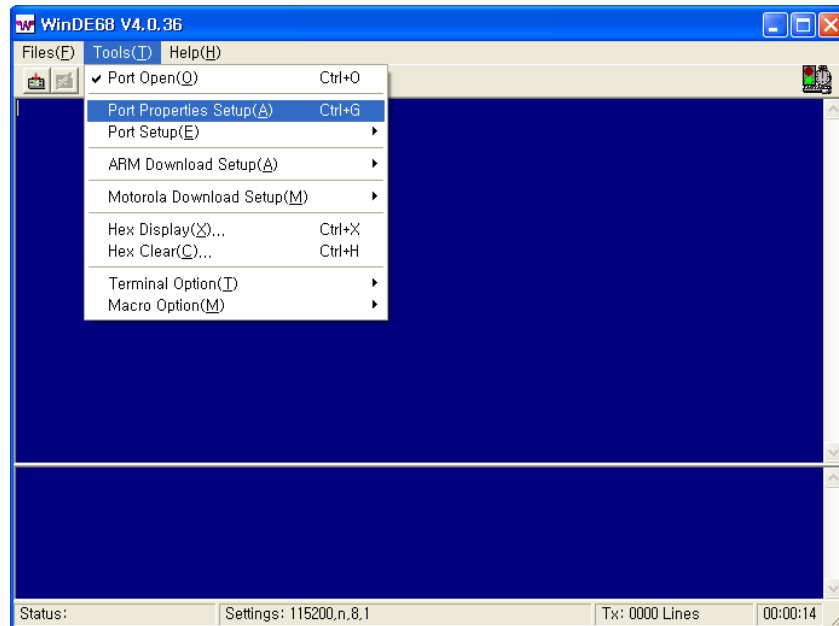


그림 13 MCU Board 와 연결되는 Com port 와 Speed 을 맞추기 위한 메뉴로 들어감.

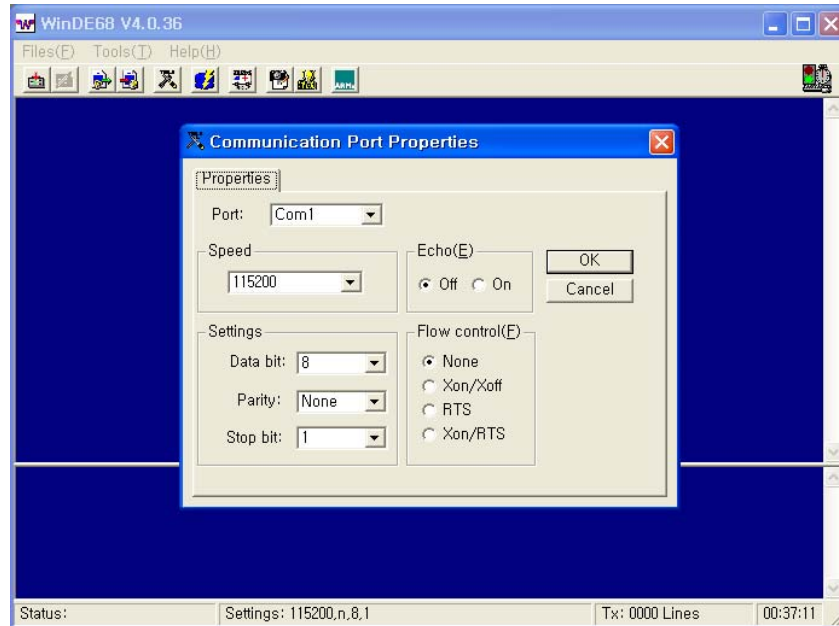


그림 14 MCU Board 와 Debug Cable 이 연결된 Com port 을 설정하고, Speed 는 115200 로 설정함.

## 5.2 Debug Cable 사양

MCU Board 와 PC 의 Com Port 을 연결하는 케이블 사양은 다음과 같습니다.

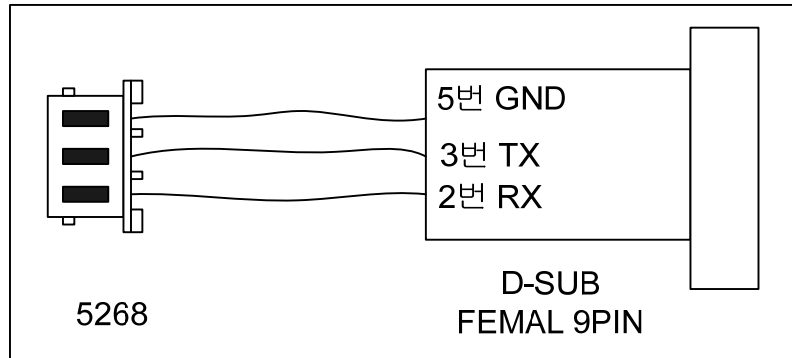


그림 15 MCU Board 에 SP3232 가 내장되어 있기 때문에 PC 의 Com port 에 바로 연결 하면 됩니다.

### 5.3 Target Board 와 연동

WINDE 와의 통신이 되려면, 먼저 STM32 MCU Board 에 WINDE 용 Bootloader Binary 을 Flash loader demonstrator (v1.3) 프로그램을 이용하여 Write 해야 합니다.

Download 완료 후에는 실제 STM32 MCU 의 Debug Port 을 사용자의 PC 와 연결 후 Comport 을 맞춘 후에 다음과 같이 Application Binary 을 다운로드 하면 됩니다.

