



Eddy DK

Programmer Guide

Ver 2.5.1.1
2014. 01. 15



Revision History

Revision Date	Document Version	Pages	Description
Feb-5-2009	2.1.0.1	All	Initial release by shlee
Sep-10-2009	2.1.0.2	All	Added Eddy-WiFi
Nov-12-2009	2.1.0.3	12	J2 pin33 PC12 → PC13 J2 pin35 PC13 → PC12
Oct-22-2009	2.1.0.3	17,18,19	J2 pin33 PC12 → PC13 J2 pin35 PC13 → PC12
		18,19	J2 pin33 J9_26 → J9_33 J2 pin34 J9_25 → J9_34 J2 pin33 J9_24 → J9_35
Nov-23-2009	2.1.0.3	2,4,6	Added S4M
Jun-25-2010	2.1.1.1	All	Open Linux Version Added Eddy-BT
Sep-15-2010	2.5.1.1	2,9	Added Eddy-CPU v2.5
Jan-20-2011	2.5.1.1		Added Eddy-S4M v2.5
Feb-15-2011	2.5.1.1		Added Eddy-CPU/mp v2.5
Aug-09-2011	2.5.1.1		Added Eddy-CPU/mp 32bit v2.5
Dec-09-2011	2.5.1.1		Added Eddy-WIFI v3.0
Jan-15-2014	2.5.1.1		Added Eddy-v2.5B(64M)

Table of Contents

1장. 개 요	5
1.1 이 매뉴얼에 대해	5
1.2 독자	5
1.3 매뉴얼 구성	6
1.4 Eddy DK 관련 문서	7
1.5 기술지원	8
2장. 시작하기	9
2.1 Eddy DK 로 무엇을 할 수 있을까?	9
2.2 Eddy DK 의 내용물	9
2.3 Eddy-CPU v2.1 / v2.5 / v2.5B	10
2.4 Eddy-DK v2.1	27
2.5 Eddy-S4M v2.1 / v2.5 / v2.5B	44
2.6 Eddy-S4M-DK v2.1	55
2.7 Eddy-S4M-JiG v2.1	65
2.8 Eddy-WiFi v3.0	69
2.9 Eddy-BT v2.1	71
2.10 Eddy-CPU/mp v2.5	73
2.11 Eddy-CPU/mp v2.5 32bit	75
3장. 개발환경 구축	77
3.1 소스코드 폴더 구조	77
3.2 사용 언어 (Language)	78
3.3 개발 환경	78
3.4 Windows 환경에 설치하기	78
3.5 Linux 환경에 설치하기	82
3.6 개발환경 제거 방법	83
4장. 응용 프로그램 컴파일	84
4.1 프로그램 종류	84
4.2 응용 프로그램 작성	87
4.3 Makefile 작성 방법	87
4.4 응용 프로그램 컴파일	88
4.5 Eddy 에서 실행하기	89
5장. Firmware 만들기	91
5.1 Firmware 를 만드는 방법	91
5.2 Firmware 업그레이드	94
6장. 라이브러리 소개	96
6.1 사용하기 전에	96
6.2 Makefile	96

6.3	System 계열 함수	96
6.4	Eddy Environment 관련 함수	97
6.5	Serial 계열 함수	99
6.6	Ethernet 계열 함수	103
6.7	GPIO Iocfl 함수	109
6.8	ADC 관련 함수	115
6.9	RTC 관련 함수	117
6.10	Debugging 관련 함수	118
7장.	Eddy Software	119
7.1	Software 구성도	119
7.2	주요 Application 설명	120
8장.	HTML 및 CGI 변경하기	121
8.1	WEB Configuration.....	121
8.2	예제 HTML 코드	121
8.3	예제 CGI 코드	123
9장.	부록	125
9.1	부트로더로 시스템 복구	125
9.2	USB 포트로 시스템 복구	130
9.3	제품 사양	140
9.4	주문 정보	149

1장. 개 요

이 장은 Eddy-DK v2.1 및 Eddy-S4M-DK v2.1 을 이용한 개발과정에 대한 소개를 담고 있다.

1.1 이 매뉴얼에 대해

본 매뉴얼은 Eddy 모듈에 적용할 수 있는 사용자 응용프로그램을 개발하는 방법과, 이를 Eddy DK (Eddy-DK, Eddy-S4M-DK) 에 저장하여 구동시키는 방법에 대해 전반적으로 소개한다. 또한, 이 과정에서 필요한 Eddy Module 에 사용되고 있는 운영체제에 대한 이해와 응용프로그램 작성에 유용하게 사용할 수 있는 API 함수들에 대한 소개도 포함한다.

이 매뉴얼을 통해 사용자가 원하는 프로그램을 작성하고, 이를 Eddy DK 를 통해 구동시킬 수 있다.

1.2 독자

본 매뉴얼은 Eddy DK 를 이용하여 새로운 응용프로그램을 개발하려는 프로그래머를 위한 것이다. 새 응용프로그램을 작성하기 전에 본 매뉴얼을 자세히 살펴보기를 권장한다.

Eddy DK 를 이용한 프로그래밍과 무관한 일반 사용자나 운영자의 경우 본 매뉴얼 대신 사용자 매뉴얼을 참고하기 바란다. 매뉴얼은 소스 파일 작성부터 Eddy module 에 업로드 하여 실행할 수 있는 Firmware 를 제작하는 방법까지의 과정을 모두 설명한다.

1.3 매뉴얼 구성

1장 개요는 본 매뉴얼에 대한 대략적인 정보 및 소개 글을 담고 있다.

2장 시작하기는 프로그래밍을 시작하기 전에 미리 준비하여야 할 일련의 과정들에 대해 소개한다.

3장 개발환경 구축은 실제로 사용자 응용프로그램을 작성하는 과정에서 필요한 개발환경을 구축하는 방법을 설명한다.

4장 응용프로그램 컴파일은 Makefile 을 통해 응용프로그램을 컴파일 하는 과정을 설명한다.

5장 Firmware 만들기는 컴파일 된 응용프로그램을 실제 모듈에 적용할 수 있도록 Firmware 파일 형태로 변환하고, 이를 Eddy 하드웨어에 적용하는 방법에 대해 설명한다.

6장 라이브러리 소개는 프로그래밍 작업에서 유용하게 사용할 수 있는 내장 라이브러리에 대해 설명한다.

7장 Eddy Software 은 본 개발 도구(Development Kit) 에 포함된 예제 응용프로그램의 동작 과정을 통해 기본적인 TCP/IP 및 시리얼 통신을 지원하는 루틴을 구현하는 방법에 대해 설명한다.

8장 HTML 및 CGI 변경하기는 Eddy DK 가 제공하는 Web 인터페이스를 사용자가 작성한 응용프로그램과 연계할 수 있도록 변경하는 방법에 대해 소개한다.

9장 부록 은 Eddy DK 를 통해 응용프로그램을 작성할 때 주의하여야 할 사항, 시스템 복구 방법 및 Firmware 업그레이드를 위한 각종 Utilities 에 대해서 설명한다.

1.4 Eddy DK 관련 문서

Eddy DK 에 관련된 기술문서는 다음과 같다.

문서명	설명
사용자 매뉴얼	Eddy 사용자를 위한 매뉴얼, Eddy 의 연결과 통신, 설정, 상태 모니터링, Firmware 업데이트, 기타 관리 작업에 대한 소개
프로그래머 가이드	프로그래머가 Eddy 에 어플리케이션을 탑재하는 데에 필요한 컴파일, 링킹, Firmware 생성 및 업로드 방법에 대한 설명 및 맞춤형 어플리케이션 제작을 위한 API 함수에 대한 설명 제공
LemonIDE 매뉴얼	Eddy 임베디드 소프트웨어 개발을 위한 Eclipse 기반의 통합개발환경 개발툴로, Windows 및 Linux 환경에서 소스의 작성 및 컴파일, 원격 디버깅, 펌웨어 생성 등을 사용법을 설명
Portview 사용자 매뉴얼	시스템베이스의 디바이스 서버 관리 프로그램인 Portview 사용 설명
COM Port Redirector 사용자 매뉴얼	시스템베이스 COM Port Redirector 사용 설명
TestView 사용자 매뉴얼	Eddy 시리얼포트와 램의 테스트 프로그램인 TestView 사용설명

Eddy나 Embedded 디바이스 서버 전반에 대한 추가정보를 얻으려면, 시스템베이스 홈페이지(<http://www.sysbas.com>)를 방문하기 바라며 홈페이지를 통해 Eddy 관련 기술문서나 최신 버전의 소프트웨어와 Firmware를 다운로드 받을 수 있다.

Eddy와 관련된 Spec Sheet 는 다음과 같다.

문서명	설명
Eddy-CPU Spec Sheet	Eddy-CPU 제품의 기술 상세
Eddy-S4M Spec Sheet	Eddy-S4M 제품의 기술 상세
Eddy-WiFi Spec Sheet	Eddy-WiFi 모듈의 기술 상세
Eddy-BT Spec Sheet	Eddy-BT 모듈의 기술 상세
LemonIDE Spec Sheet	통합개발환경의 기술 상세
Eddy White Paper	디바이스서버 일반에 대한 개괄, 배경과 기술 설명/시장 환경

모든 문서는 항상 최신 버전으로 업데이트 하여 홈페이지에 게재된다. 문서의 내용은 사전 공지 없이 변경될 수 있음을 알린다.

1.5 기술지원

시스템베이스는 다음의 세 가지 방법으로 고객에 대한 기술지원을 제공한다.

- 1) 당사 홈페이지 <http://www.sysbas.com/>의 Support/기술지원 또는 기술지원 전용 홈페이지 <http://www.solveonline.com/>을 방문하면 자주 묻는 질문(FAQ)이나 게시판을 통해 기술 지원을 받을 수 있다.
- 2) 시스템베이스의 기술팀(tech@sysbas.com)으로 e-mail을 보내면 빠른 시간에 답변을 받을 수 있다. 어떠한 질문, 요청, 의견도 좋다.
- 3) 보다 빠른 기술 지원을 받기 원한다면 전화를 통한 고객 상담을 받으실 수 있다. 시스템베이스의 기술팀에서는 고객의 어떤 어려운 문제라도 친절하게 상담과 해결 방법을 지원하고 있다. 전화번호는 02-855-0501이다.

Copyright 2009-2014 SystemBase Co., Ltd. All rights reserved.

Website <http://www.sysbas.com/>

Tel 02-855-0501

Fax 02-855-0580

서울시 구로구 디지털로 288. 대륭포스트타워 1 차 1601 호

문의 사항에 대해서는 tech@sysbas.com으로 연락바랍니다.

2장. 시작하기

이 장에서는 Eddy DK 의 개요와 핵심 기능, 패키지 구성과 활용 분야에 대해 설명한다.

2.1 Eddy DK 로 무엇을 할 수 있을까?

Eddy DK 는 Eddy Module 을 사용자의 하드웨어에 적용할 수 있는 응용프로그램을 쉽고 빠르게 개발할 수 있도록 도와주기 위해 만들어진 제품이다. 기존에 새 하드웨어를 만들고 이에 대한 운영체제를 확보하였다. 따라서 여기에 쓰이는 응용 프로그램을 모두 직접 제작하는 것이 커다란 부담이었다. Eddy DK 는 이러한 어려움을 경감시킬 수 있다.

Eddy DK 제품은 사용자가 직접 작성한 애플리케이션을 업로드 및 실행할 수 있다는 점에서 다른 디바이스 서버에 비해 차별성을 가지고 있다. 이러한 기능으로 인해 사용자는 Linux 환경에서 작성한 기존의 대부분 소켓/시리얼 통신 애플리케이션을 별다른 수정 없이 직접 모듈에 업로드 할 수 있다. 이러한 개방성으로 인해 사용자는 폭넓고 다양한 기능을 상대적으로 적은 제한을 받으며 적용시킬 수 있다.

Eddy DK 는 개발자들의 애플리케이션을 모듈 상에서 직접 실행할 수 있도록 하기 위해 IDE(LemonIDE) 와 SDK 환경을 지원한다. 프로그래머는 Windows 및 Linux 환경에서 SDK 와 실행 가능한 예제 코드를 이용하여 자신만의 애플리케이션을 쉽게 작성할 수 있다. Linux 에서 실행되는 크로스 컴파일러는 애플리케이션이 Eddy 모듈에서 원활하게 실행되도록 도움을 준다. Embedded Linux 운영체제로 인해 애플리케이션을 더 안정된 상태에서 더 빠르게 실행 가능하다.

2.2 Eddy DK 의 내용물

Eddy DK 를 구매 하면 기본적으로 Eddy module 이 포함된다.

Eddy DK 패키지 구성은 아래와 같다. 모든 구성품이 포함되어 있는지 확인하기 바란다.

Eddy-DK 의 경우 (Eddy-CPU v2.1 또는 v2.5 1개, Eddy-DK v2.1보드 1개)

Eddy-S4M-DK 의 경우 (Eddy-S4M v2.1 또는 v2.5 1개, Eddy-S4M-DK 보드 1개, (Option : Eddy-S4M-JIG))

시리얼 케이블 1개

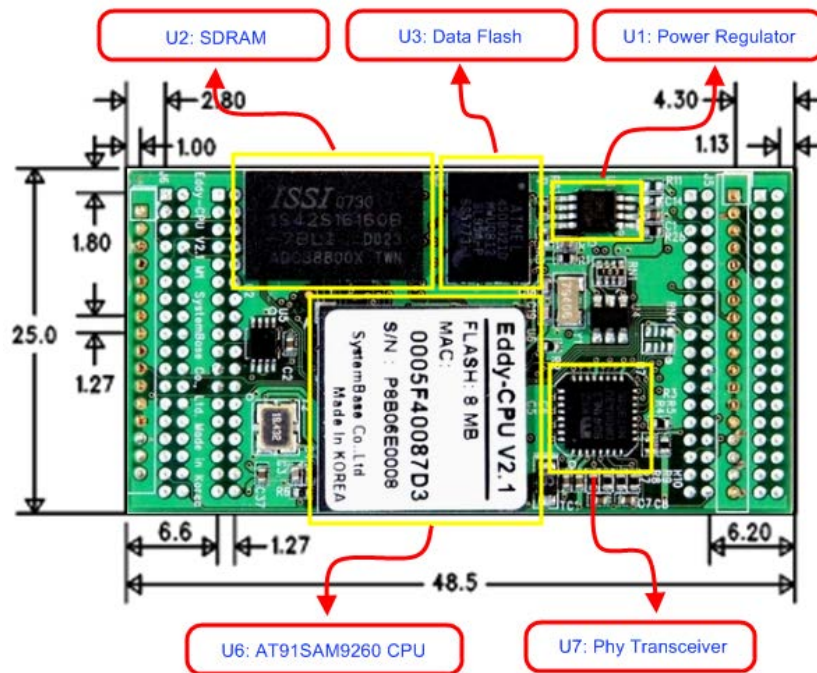
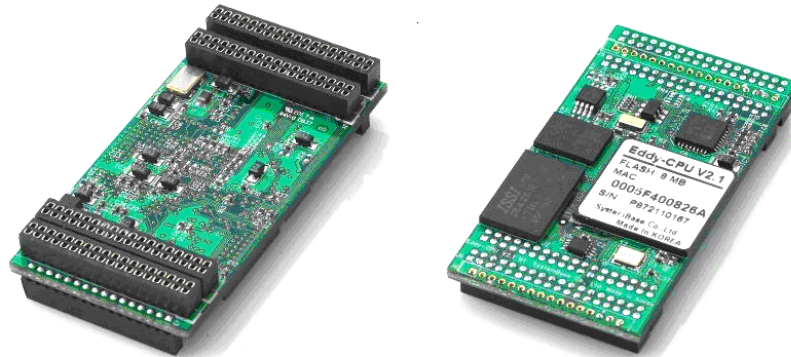
LAN 케이블 1개

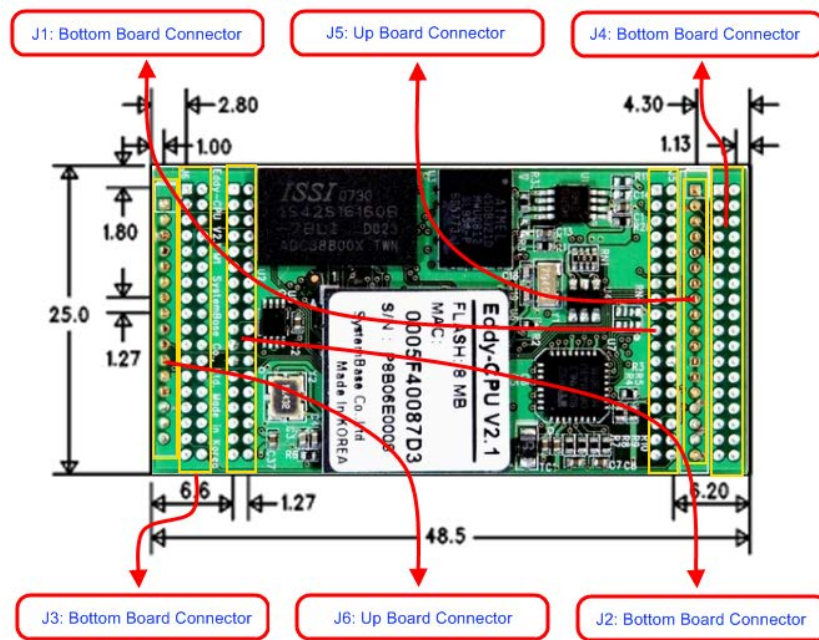
USB A to B 케이블 1개

전원 어댑터 1개

CD (시스템베이스 SDK, LemonIDE, 컴파일 환경, 유틸리티, 각종 매뉴얼)

2.3 Eddy-CPU v2.1 / v2.5 / v2.5B





* Eddy-CPU v2.1 / v2.5 Pin Assignment

J1			
Pin	Signal Name	Pin	Signal Name
1	PA5	2	PA4
3	PC5	4	PC19
5	PC21	5	PC23
7	HDMA	8	NC
9	HDPA	10	DDM
11	PC26	12	DDP
13	PC4 (RDY#)	14	PC16
15	ICE_NTRST	16	RTCK
17	TDO	18	TMS
19	TDI	20	TCK
21	3.3V	22	GND
23	3.3V	24	GND
25	PB29 (CTS1)	26	PB28 (RTS1)
27	PB6 (TXD1)	28	PB7 (RXD1)
29	A20	30	A19
31	LAN_Speed	32	LAN_Link
33	LAN_RX-	34	LAN_RX+
35	LAN_TX-	36	LAN_TX+

J2			
Pin	Signal Name	Pin	Signal Name
1	A15	2	A14
3	A13	4	A12
5	A11	5	A10
7	A9	8	A8
9	A7	10	A6
11	A5	12	A4
13	A3	14	A2
15	A1	16	A0
17	PC9	18	NWE
19	FPG	20	NRD
21	GND	22	3.3V
23	GND	24	3.3V
25	D7	26	D6
27	D5	28	D4
29	D3	30	D2
31	D1	32	D0
33	PC13	34	JTAGSEL
35	PC12	36	NC

J3			
Pin	Signal Name	Pin	Signal Name
1	PID0	2	PID1
3	PID2	4	PID3
5	PID4	5	GND
7	PC14	8	PC17
9	PC18	10	PC8 (RTS3)
11	PC20	12	PC10 (CTS3)
13	PA22	14	PC15 (IRQ1)
15	PB8	16	PB9 (RXD2)
17	PB10	18	PB11(RXD3)
19	PC0	20	PC1 (AD1)
21	PC2	22	PC3 (AD3)
23	PB14 (DRXD)	24	PB15 (DTXD)
25	GND	26	GND
27	BMS	28	NRST
29	PB23 / DCD0	30	PB5 / RXD0
31	PB4 / TXD0	32	PB24 / DTR0
33	PB22 / DSR0	34	PB26 / RTS0
35	PB27 / CTS0	36	PB25 / RI0

J4			
Pin	Signal Name	Pin	Signal Name
1	PB12	2	PB13
3	PB30	4	PB31
5	PB0	5	PC22
7	PB1	8	PB16
9	PB2	10	PB17
11	PB3	12	PB18
13	BHDM	14	PB19
15	BHDP	16	PB20
17	A16	18	PB21
19	A17	20	A18
21	D8	22	D9
23	D10	24	D11
25	D12	26	D13
27	D14	28	D15
29	TWD	30	TCK
31	NANDOE	32	NAND_CLE /
33	NANDWE	34	NAND_ALE /
35	NC	36	NC

J5	
Pin	Signal Name
1	PB0
2	PB1
3	PB2
4	PB3
5	3.3V
6	3.3V
7	BHDM, USB Host Data(-)
8	BHDP, USB Host Data(+)
9	PA31 / TXD4
10	PA30 / RXD4
11	NRST
12	GND
13	GND
14	PA9 / WPID0
15	PC6 / WPID1
16	PC7 / WPID2
17	NC
18	NC

J6	
Pin	Signal Name
1	NC
2	NC
3	3.3V
4	3.3V
5	PC25 / BT_Factory
6	PB10 / TXD3
7	PB11 / RXD3
8	PC8 / RTS3
9	PC10 / CTS3
10	PC24 / BT_MODE
11	NRST
12	GND
13	GND
14	NC
15	NC
16	NC

J1 Specifications

J1			
Pin	Signal Name	Pin	Signal Name
1	PA5	2	PA4
3	PC5	4	PC19
5	PC21	5	PC23
7	HDMA	8	NC
9	HDP A	10	DDM
11	PC26	12	DDP
13	PC4 (RDY#)	14	PC16 (nRESET)
15	ICE NTRST	16	RTCK
17	TDO	18	TMS
19	TDI	20	TCK
21	3.3V	22	GND
23	3.3V	24	GND
25	PB29 (CTS1)	26	PB28 (RTS1)
27	PB6 (TXD1)	28	PB7 (RXD1)
29	A20	30	A19
31	LAN Speed	32	LAN Link
33	LAN RX-	34	LAN RX+
35	LAN_TX-	36	LAN_TX+

J1 Pin Description

Pin No	Name	DK v2.1 Pin No	Expansion Header Pin No	Description	
1	PA5	J10_1	J4_2	Peripheral A : CTS2	UART #2 Clear to Send Signal
				Peripheral B : MCBD1	사용불가. Eddy-CPU v2.1/ v2.5 /v2.5B 에서는 SPI0 와 연결된Data Flash을 사용한다. 따라서SPI0 와 다중(multiplexing)으로 사용되는 MCDB0, MCDB3, MCCDB 신호를 사용할 수 없어 Multimedia Card Slot B 사용이 불가능하다.
2	PA4	J10_2	J4_1	Peripheral A : RTS2	UART #2 Request to Send Signal
				Peripheral B : MCDB2	사용불가.
3	PC5	J10_3	J4_12	Peripheral A : A24	External Address Bus
				Peripheral B : SPI1_NPCS1	SPI1(Serial Peripheral Interface) Peripheral Chip Select 1
4	PC19	J10_4	J4_24	Peripheral A : A24	Multimedia Card Slot B Data
				Peripheral B : SPI1_NPCS2	SPI1(Serial Peripheral Interface) Peripheral Chip Select 2
5	PC21	J10_5	J4_26	Peripheral A : D21	External Data bus
				Peripheral B : EF100	Ethernet(WAN) Force 100Mbit/sec.
6	PC23	J10_6	J4_28	Peripheral A : D23	External Data Bus
7	HDMA	J10_7	J1_27	USB Host Port A Data -	
8	NC	J10_8	--	Not Connect	
9	HDPA	J10_9	J1_29	USB Host Port A Data +	
10	DDM	J10_10	-	USB Device Port Data -	

11	PC26	J10_11	-	D26	External Data Bus
12	DDP	J10_12	-	USB Device Port Data +	
13	PC4 (RDY#)	J10_13	J4_11	Eddy DK v2,1 : RDY#(OUT)	Ready 신호. Output signal for CPU operation status
				Peripheral A : A23	External Address Bus
				Peripheral B : SPI1_NPCS2	SPI1(Serial Peripheral Interface) Peripheral Chip Select 2
14	PC16 (nRESET)	J10_14	J4_21	Eddy DK v2,1 : nRESET#(IN)	외부 Reset key로 부터 입력 신호를 지속적으로 polling 하고, "Low"의 지속시간을 check 하여 S/W 적으로 아래와 같이 구현해 야 한다. 5 초미만 : 일반적인 reset 기능 5 초이상 : Factory Default 기능
				Peripheral A : D16	External Data Bus
				Peripheral B : SPI0_NPCS2	사용불가 SPI0 의 SPI0_SPCK, SPI0_MISO, SPI0_MOSI 신호는 외부로 연결 되지 않아 사용할 수 없다.
15	ICE_NTRST	J10_15	J7_3	ICE Test Reset Signal	
16	RTCK	J10_16	J7_11	Return Test Clock	
17	TDO	J10_17	J7_13	Test Data Out	
18	TMS	J10_18	J7_7	Test Mode Select	
19	TDI	J10_19	J7_5	Test Data In	
20	TCK	J10_20	J7_9	Test Clock	

21	3.3V	3.0V to 3.6V power input				
22	GND	Ground				
23	3.3V	3.0V to 3.6V power input				
24	GND	Ground				
25	PB29	J10_25	J2_30	Peripheral A : CTS1	USART1 Clear To Send	
				Peripheral B : ISI_VSYNC	Image Sensor Vertical Synchronization	
26	PB28	J10_26	J2_29	Peripheral A : RTS1	USART1 Request To Send	
				Peripheral B : ISI_PCK (IN)	Image Sensor Pixel Clock Provided by the Image Sensor	
27	PB6	J10_27	J2_7	Peripheral A : TXD1	USART1 Transmit Data	
				Peripheral B : TCLK1	Timer Counter ch1 External CLK IN	
28	PB7	J10_28	J2_8	Peripheral A : RXD11	USART1 Receive Data	
				Peripheral B : TCLK2	Timer Counter ch2 External CLK IN	
Address Bus						
29	A20	J10-29	J1_31	Address Bus		
30	A19	J10_30	J1_32	Address Bus		
Ethernet 10/100 (Auto MDI/MDIX)						
31	LED_Speed	J10_31	-	LAN connection speed		
				Speed	Pin State	LED Definition
				10Base-T	H	OFF
				100Base-TX	L	ON
32	LED_Link	J10_32	-	LAN connection status		
				Link/Activity	Pin State	LED Definition
				No Link	H	OFF
				Link	L	ON

				Activity	Toggle	Blinking	
33	LAN_RX-	J10_33	-	CPU 내부 Ethernet PHY(WAN)의 Physical receive or transmit signal (- differential)			
34	LAN_RX+	J10_34	-	CPU 내부 Ethernet PHY(WAN)의 Physical receive or transmit signal (+ differential)			
35	LAN_TX-	J10_35	-	CPU 내부 Ethernet PHY(WAN)의 Physical transmit or receive signal (- differential)			
36	LAN_TX+	J10_36	-	CPU 내부 Ethernet PHY(WAN)의 Physical transmit or receive signal (+ differential)			

J2 Specifications

Connect USB cable to J1 while the jumper is connected to J2, so that applications can be compiled, linked, created, and uploaded to the Eddy-CPU module. (Please refer to Programmer Guide for more information.)

J2			
Pin	Signal Name	Pin	Signal Name
1	A15	2	A14
3	A13	4	A12
5	A11	5	A10
7	A9	8	A8
9	A7	10	A6
11	A5	12	A4
13	A3	14	A2
15	A1	16	A0
17	PC9	18	NWE
19	FPG	20	NRD
21	GND	22	3.3V
23	GND	24	3.3V
25	D7	26	D6
27	D5	28	D4
29	D3	30	D2
31	D1	32	D0
33	PC12	34	JTAGSEL
35	PC13	36	NC

J2 Pin Description

Pin No	Name	DK v2.1 Pin No	Expansion Header Pin No	Description
1~16	A[15:0]	J9_1 ~J9_16	J3_4-J3_20	External Address Bus 0-15 (0 at reset) CPU와 DK는 직접 연결되고 확장 커넥터(J3)와는 버퍼를 통해 연결된다.
17	PC9	J9_17	J4_14	Peripheral A : NCS5 External device Chip Select 5, 256MB memory area addressable, active low
				Peripheral B : TIOB0 Timer Counter ch0 I/O Line B
18	NEW	J9_18	J1_21	External device Write Enable signal, active low
19	FPG	J9_19	-	For Flash Programming USB 포트를 통해서 Eddy-CPU v2.1/V2.5/V2.5B 의 Data Flash 에 부트코드(로더, 커널, 파일시스템)를 프로그래밍하기 위해 사용된다. 자세한 사항은 2.4.2.3 S6:NAND Flash & Data Flash Chip Select참조.
20	NRD	J9_20	J1_23	External device Read Enable signal, active low
21, 23	GND	Ground		
22, 24	3.3V	3.0V to 3.6V power input		
25~32	D[7:0]	J9_25 - J3_32	J3_29 - J3_36	External Data Bus 0-7 CPU와 DK는 직접 연결되고 확장 커넥터(J3)와는 버퍼를 통해서 연결된다. PC13(NCS6 : Chip Select 6)을 enable 시켜야 버퍼가 동작되고, Reset시에는 Pulled-up input 이다.
33	PC12	J9_24	J4_17	Peripheral A : IRQ0 External Interrupt Input 0
				Peripheral B : NCS7 External device Chip Select 7, 256MB memory area addressable, active low
34	JTAGSEL	J9_25	-	JTAG boundary scan can be used by connecting pin34 and 36(J14 연결). This pin should not be connected when using ICE(In-Circuit Emulator) or in normal operation status.

35	PC13	J9_26	J4_18	Edd-DK v2.1 : NCS6	확장 Header에 연결된 Data Bus는 NCS6 을 enable 해야 사용할 수 있다.
				Peripheral A : FIQ	Fast Interrupt Input
				Peripheral B : NCS6	External device Chip Select 6 256MB memory area addressable, active low
36	NC	Not Connect			

J3 Specifications

J3			
Pin	Signal Name	Pin	Signal Name
1	PID0	2	PID1
3	PID2	4	PID3
5	PID4	5	GND
7	PC14	8	PC17
9	PC18	10	PC8 (RTS3)
11	PC20	12	PC10 (CTS3)
13	PA22	14	PC15 (IRQ1)
15	PB8	16	PB9 (RXD2)
17	PB10	18	PB11(RXD3)
19	PC0	20	PC1 (AD1)
21	PC2	22	PC3 (AD3)
23	PB14 (DRXD)	24	PB15 (DTXD)
25	GND	26	GND
27	BMS	28	NRST
29	PB23 / DCD0	30	PB5 / RXD0
31	PB4 / TXD0	32	PB24 / DTR0
33	PB22 / DSR0	34	PB26 / RTS0
35	PB27 / CTS0	36	PB25 / RI0

J3 Pin Description

Pin No	Name	DK v2.1 Pin No	Expansion Header Pin No	Description	
1-5	PID[4:0]	J8_1 ~J8_5	-	Product ID only used by the manufacturer. Please do not work on these pins.	
6,25,26	GND	Ground			
7	PC14	J8_7	J4_19	Peripheral A : NCS3	External Device Chip Select 3
				Peripheral B : IRQ2	External Interrupt Input 2
8	PC17	J8_8	J4_22	Peripheral A : D17	External Data Bus
				Peripheral B : SPI0_NPCS3	사용불가
9	PC18	J8_9	J4_23	Peripheral A : D18	External Data Bus
				Peripheral B : SPI1_NPCS1	SPI1(Serial Peripheral Interface) Peripheral Chip Select 1
10	PC8	J8_10	J4_13	Peripheral A : NCS4	External Device Chip Select 4
				Peripheral B : RTS3	USART3 Request to Send
11	PC20	J8_11	J4_25	Peripheral A : D20	External Data Bus
				Peripheral B : SPI1_NPCS3	SPI1(Serial Peripheral Interface) Peripheral Chip Select 3
12	PC10	J8_12	J4_15	Peripheral A : A25	External Address Bus
				Peripheral B : CTS3	USART3 Clear to Send
13	PA22	J8_13	-	Digital I/O Input 4	

14	PC15	J8_14	J4_20	Peripheral A : NWAIT	External Wait Signal Input
				Peripheral B : IRQ1	External Interrupt Input 2
15	PB8	J8_15	J2_9	Peripheral A : TXD2	UART2 Transmit Data
16	PB9	J8_16	J2_10	Peripheral A : RXD2	UART2 Receive Data
17	PB10	J8_17	J2_11	Peripheral A : TXD3	UART3 Transmit Data
				Peripheral B : ISI_D8	Image Sensor Data 8
18	PB11	J8_18	J2_12	Peripheral A : RXD3	UART3 Receive Data
				Peripheral B : ISI_D9	Image Sensor Data 9
19	PC0	J8_19	J4_7	Peripheral A : AD0	Analog to Digital Converter Input Ch0
				Peripheral B : SCK3	USART3 Serial Clock
20	PC1	J8_20	J4_8	Peripheral A : AD1	Analog to Digital Converter Input Ch1
				Peripheral B : PCK0	Programmable Clock Output 0
21	PC2	J8_21	J4_9	Peripheral A : AD2	Analog to Digital Converter Input Ch2
				Peripheral B : PCK1	Programmable Clock Output 1
22	PC3	J8_22	J4_10	Peripheral A : AD3	Analog to Digital Converter Input Ch3
				Peripheral B : SPI1_NPCS3	SPI1(Serial Peripheral Interface) Peripheral Chip Select 3
23	PB14	J8_23	J2_15	Peripheral A : DRXD	Debug Receive Data
24	PB15	J8_24	J2_16	Peripheral A : DTXD	Debug Transmit Data
27	BMS	J8_27	-	Boot Mode Select signal BMS = 1, Boot on Embedded ROM	

				BMS = 0, Boot on External Memory	
28	NRST	J8_28	J1_20	External device Reset signal, active low signal	
29	PB23	J8_29	J4_28	Peripheral A : DCD0	USART0 Data Carrier Detection
				Peripheral B : ISI_D3	Image Sensor Data 3
30	PB5	J8_30	J2_6	Peripheral A : RXD0	USART0 Receive Data
31	PB4	J8_31	J2_5	Peripheral A : TXD0	USART0 Transmit Data
32	PB24	J8_32	J2_25	Peripheral A : DTR0	USART0 Data Terminal Ready
				Peripheral B : ISI_D4	Image Sensor Data 4
33	PB22	J8_33	J2_23	Peripheral A : DSR0	USART0 Data Set Ready
				Peripheral B : ISI_D2	Image Sensor Data 2
34	PB26	J8_34	J2_27	Peripheral A : RTS0	USART0 Request To Send
				Peripheral B : ISI_D6	Image Sensor Data 6
35	PB27	J8_35	J2_28	Peripheral A : CTS0	USART0 Clear To Send
				Peripheral B : ISI_D7	Image Sensor Data 7
36	PB25	J8_36	J2_26	Peripheral A : RI0	USART0 Ring Indicator
				Peripheral B : ISI_D5	Image Sensor Data 5

J4 Specifications

J4			
Pin	Signal Name	Pin	Signal Name
1	PB12	2	PB13
3	PB30	4	PB31
5	PB0	5	PC22
7	PB1	8	PB16
9	PB2	10	PB17
11	PB3	12	PB18
13	BHDM	14	PB19
15	BHDP	16	PB20
17	A16	18	PB21
19	A17	20	A18
21	D8	22	D9
23	D10	24	D11
25	D12	26	D13
27	D14	28	D15
29	TWD	30	TCK
31	NANDOE	32	NAND CLE /
33	NANDWE	34	NAND ALE /
35	NC	36	NC

J4 Pin Description

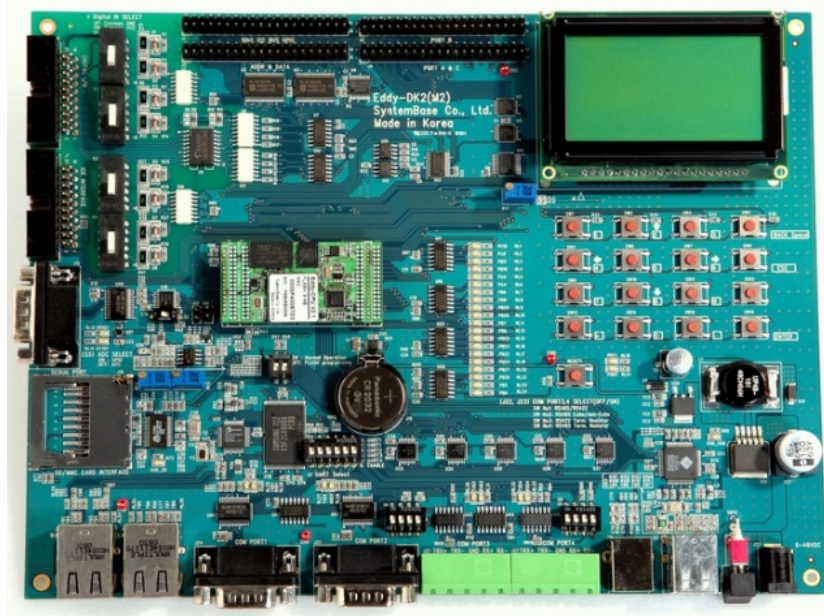
Pin No	Name	DK v2.1 Pin No	Expansion Header Pin No	Description	
1	PB12	J11_1	J2_17	Peripheral A : TXD5	USART5 Transmit Data
				Peripheral B : ISI_D10	Image Sensor Data 10
2	PB13	J11_2	J2_18	Peripheral A : RXD5	USART5 Receive Data
				Peripheral B : ISI_D11	Image Sensor Data 11
3	PB30	J11_3	J2_31	Peripheral A : PCK0	Programmable Clock Output 0
				Peripheral B : ISI_HSYNC	Image Sensor Horizontal Synchronization
4	PB31	J11_4	J2_32	Peripheral A : PCK1	Programmable Clock Output 1
5	PB0	J11_5	J2_2	Peripheral A : SPI1_MISO	SPI1(Serial Peripheral Interface) Master In Slave Out
				Peripheral B : TIOA3	Timer Counter ch3 I/O Line A
6	PC22	J11_6	J4_27	Peripheral A : D22	
				Peripheral B : TCLK5	Timer Counter ch5 External CLK IN
7	PB1	J11_7	J2_3	Peripheral A : SPI1_MOSI	
				Peripheral B : TIOB3	Timer Counter ch3 I/O Line B
8	PB16	J11_8	J2_17	Peripheral A : TK0	SSC Transmit Clock
				Peripheral B : TCLK3	Timer Counter ch3 External CLK IN
9	PB2	J11_9	J2_4	Peripheral A : SPI1_SPCK	SPI1(Serial Peripheral Interface) Serial Clock

				Peripheral B : ISI_D3	Image Sensor Data 3
10	PB17	J11_10	J2_18	Peripheral A : TF0	SSC Transmit Frame Sync
				Peripheral B : TCLK4	Timer Counter ch4 External CLK IN
11	PB3	J11_11	J2_5	Peripheral A : SPI1_NPCS0	SPI1(Serial Peripheral Interface) Peripheral Chip Select 0
				Peripheral B : TIOA5	Timer Counter ch5 I/O Line A
12	PB18	J11_12	J2_19	Peripheral A : TD0	SSC Transmit Data
				Peripheral B : TIOB4	Timer Counter ch4 I/O Line B
13	HDMB	J11_13	J1_28	USB Device Port Data -	
14	PB19	J11_14	J2_20	Peripheral A : RD0	SSC Receive Data
				Peripheral B : TIOB5	Timer Counter ch5 I/O Line B
15	HDPB	J11_15	J1_30	USB Device Port Data +	
16	PB20	J11_16	J2_21	Peripheral A : RK0	SSC Receive Clock
				Peripheral B : ISI_D0	Image Sensor Data 0
17	A16	J11_17	J3_3	External Address Bus	
18	PB21	J11_18	J2_22	Peripheral A : RF0	SSC Receive Frame Sync
				Peripheral B : ISI_D1	Image Sensor Data 1
19	A17	J11_19	J3_2	External Address Bus	
20	A18	J11_20	J3_1		
21-28	D[8:15]	J11_21	J3_28	External Data Bus 8-15	

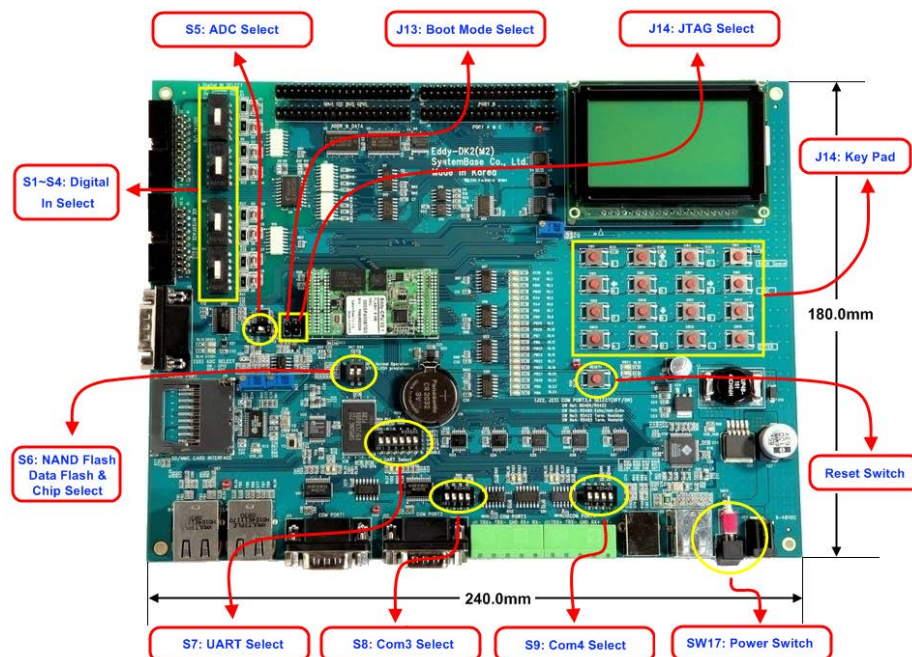
		~J11_28	~J3_21	CPU와 DK는 직접 연결되고 확장 커넥터(J3)와는 버퍼를 통해서 연결된다. PC13(NCS6 : Chip Select 6)을 enable 시켜야 버퍼가 동작 되고, Reset시는 Pulled-up input 으로 동작한다.
29	TWD	J11_29	J4_3	Two-wire Serial Data. 이 핀은 GPIO로 사용할 수 없다.
30	TWCK	J11_30	J4_4	Two-wire Serial Data. 이 핀은 GPIO로 사용할 수 없다.
31	NANDOE	J11_31	-	NAND Flash Output Enable
32	A22	J11_32	J1_29	Address Bus CPU와 DK는 직접 연결되고 확장 커넥터(J3)와는 버퍼를 통해서 연결된다.
33	NANDWE	J11_33	-	NAND Flash Write Enable
34	A21	J11_34	J1_30	Address Bus
35,36	NC	Not Connect		

2.4 Eddy-DK v2.1

2.4.1 제품별 위치 (Modules' Locations)



2.4.2 스위치 설명 (Switch Description)



2.4.2.1. S1~S4: Digital In Select

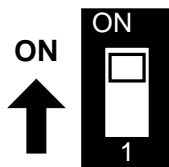
스위치 S1~S4를 설정하여 Digital Input을 2가지 모드로 선택하는 것이 가능하다. 연결되는 장치가 VCC Common Mode로 동작하는지 GND Common Mode로 동작하는지에 아래와 같이 Switch를 설정하면 된다. DK내 Digital input 회로는 참조를 위한 회로이므로 실제 사용 시에는 전압 및 전류 허용치를 참고하여 설계하여야 한다.

Common 입력 설정 (S1~S4 공통)

MODE	Switch	설명
GND Common	UP ON	
VCC Common	Down ON	

2.4.2.2. S5: ADC Select

PC0-PC4를 어떻게 사용할 것인지를 결정하는 스위치이다. DK 내 온도센서와 조도센서를 통해 Analog 입력을 받을 것인지 아니면 Expansion Header를 통해 연결되는 GPIO로 사용할지를 선택할 수 있다.



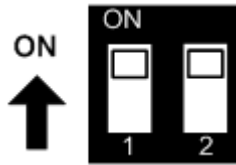
SW Off : ADC 사용
SW ON : GPIO 사용

핀이름	기 능	용 도	I/O
PC0	ADC0	온도 센서 입력(LM50), RN: U22	IN
PC1	ADC1	조도 센서 입력(BH1600), RN: U26	IN
PC2	ADC2	온도 센서 입력(TMP300), RN: U24	IN
PC3	ADC3	N/A	IN

* 여기서 RN = Reference Number

2.4.2.3. S6:NAND Flash & Data Flash Chip Select

USB device 포트를 통한 Flash programming 또는 CPU 내 Data Flash와 NAND Flash 중 어떤 것을 사용하여 부팅을 할 것인가를 선택할 수 있다.

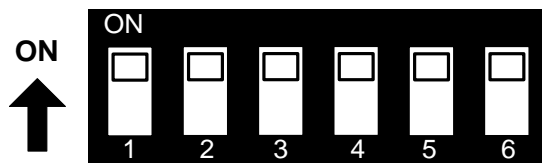


Flash Programming & 부팅 device 선택		
Switch No 1	Switch No 2	Operation description
OFF	OFF	For Flash Programming USB 포트를 통해서 Eddy-CPU v2.1/V2.5/v2.5B 의 Data Flash 에 부트코드 (로더, 커널, 파일시스템) 를 프로그래밍 한다.
OFF	ON	Eddy-CPU v2.1/v2.5/v2.5B 의 Data Flash 를 통해서 부팅한다.
ON	OFF	Eddy DK v2.1 보드상의 NAND Flash 를 통해서 부팅한다.
ON	ON	Data Flash 또는 NAND flash 를 통해서 부팅하도록 한다. Data Flash 와 NAND Flash 둘다 Boot programming 되었다면 CPU Boot program algorithm 에 따라 SPI 에 연결된 Data Flash 가 실행된다. 만약 Data Flash 에서 유효한 ARM vector sequence 가 발견하지 못하면 그때는 NAND flash Boot program 이 실행될 것이다. (Datasheet 13 장 AT91SAM9260 Boot Program 참조)

2.4.2.4. S7:UART Select

스위치를 Off 상태로 설정하면, UART와 Serial driver와 연결되어 Serial Port 시험이 가능하다.

스위치를 ON으로 설정하고 각 포트를 GPIO로 설정하게 되면 해당 GPIO와 연결된 LED를 통해 제어상태를 확인할 수 있다. 만약 Switch No.6을 ON(UP)시키면 UART 4포트 전체가 Serial driver 와 GPIO LED 어느 것과도 연결되지 않고 오직 Expansion Header와만 연결되어 사용자 임의로 사용이 가능하다.

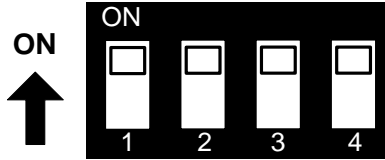


Serial Port & LED 제어			
Switch Bank	Switch No	Down Position(OFF) Serial Port Test	UP Position(ON) GPIO TEST (High : LED On)
S7	1	UART#0 TEST UART#0 의 TXD, RXD, RTS, CTS 신	RS232 Driver 와 연결이 단절되고, GPIO (PB4, PB5, PB26, PB27)로 설정

		호가 RS232 driver 에 연결된다.	하면 LED 제어 가능.
2	UART#0 TEST UART#0 의 DTR, DSR, DCD, RI 신호가 RS232 driver 에 연결된다.	RS232 Driver 와 연결이 단절되고, GPIO (PB24, PB22, PB23, PB25)로 설정되면 LED 제어 가능.	
3	UART#1 TEST UART#1 의 TXD, RXD, RTS, CTS 신호가 RS232 driver 에 연결된다.	RS232 Driver 와 연결이 단절되고, GPIO (PB6, PB7, PB28, PB29)로 설정되면 LED 제어 가능.	
4	UART#2 TEST UART#2 의 TXD, RXD, RTS, CTS 신호가 RS422/485 driver 에 연결된다.	RS422/485 Driver 와 연결이 단절되고, GPIO (PB8, PB9, PA4, PA5)로 설정되면 LED 제어 가능.	
5	UART#3 TEST UART#3 의 TXD, RXD, RTS, CTS 신호가 RS422/485 driver 에 연결된다.	RS422/485 Driver 와 연결이 단절되고, GPIO (PB10, PB11, PC8, PC10)로 설정되면 LED 제어 가능.	
6	For Serial Port & GPIO Test Serial Port 와 DK 보드 내 GPIO LED 를 시험하기 위해서는 항상 OFF 상태를 유지해야 한다.	Connect to Expansion Header UART#0~#3 의 모든 신호가 시리얼 드라이버와 GPIO LED 로 연결되지 않고 Expansion Header(J2, J4)와 직접 연결된다.	

2.4.2.5. S8:COM3 & S9: COM4 Select

COM Port #3 과 COM Port #4는 RS422/RS485를 지원하는 Combo 포트이다. 이들 포트에서 RS422/RS485 선택 및 RS485 echo 또는 non-echo mode 등의 설정은 각 포트옆에 위치한 스위치를 통해서 하드웨어적으로 이루어진다.



COM PORT#3, #4 settings			
Switch Bank	Switch No	Down Position(OFF)	UP Position(ON)
S8 Port#3	1	RS485 Half-Duplex 설정	RS422 Full-Duplex 설정
	2	RS422(RX enabled) RS485 echo-mode	RS485 non echo-mode
	3	RS422 Termination Resistor not connected	RS422 Termination Resistor Connected
	4	RS485 Termination Resistor not connected	RS422 Termination Resistor Connected
S9 Port#4	1	RS485 Half-Duplex	RS422 Full-Duplex
	2	RS422(RX enabled) RS485 echo-mode	RS485 non echo-mode
	3	RS422 Termination Resistor not connected	RS422 Termination Resistor Connected
	4	RS485 Termination Resistor not connected	RS422 Termination Resistor Connected

2.4.2.6. SW1~SW16: Key Pad

DK 내 Key Pad는 4x4 matrix를 사용하도록 설계되었다. 이는 GPIOs를 입력으로 설정하여 Key 값을 읽어들이는. 또한 Key 2, 4, 6, 8은 LCD 메뉴 선택을 위한 ▲(UP), ▼(DN), ◀(LEFT), ▶(RIGHT) 방향키로 사용가능 하다.

GPIOs	4x4 Key matrix 와 연결	I/O
PB20	첫번째 Row 연결	IN
PB21	두번째 Row 연결	IN
PB30	세번째 Row 연결	IN
PB31	네번째 Row 연결	IN

PC20	첫번째 Column 연결	IN
PC21	두번째 Column 연결	IN
PC22	세번째 Column 연결	IN
PC23	네번째 Column 연결	IN

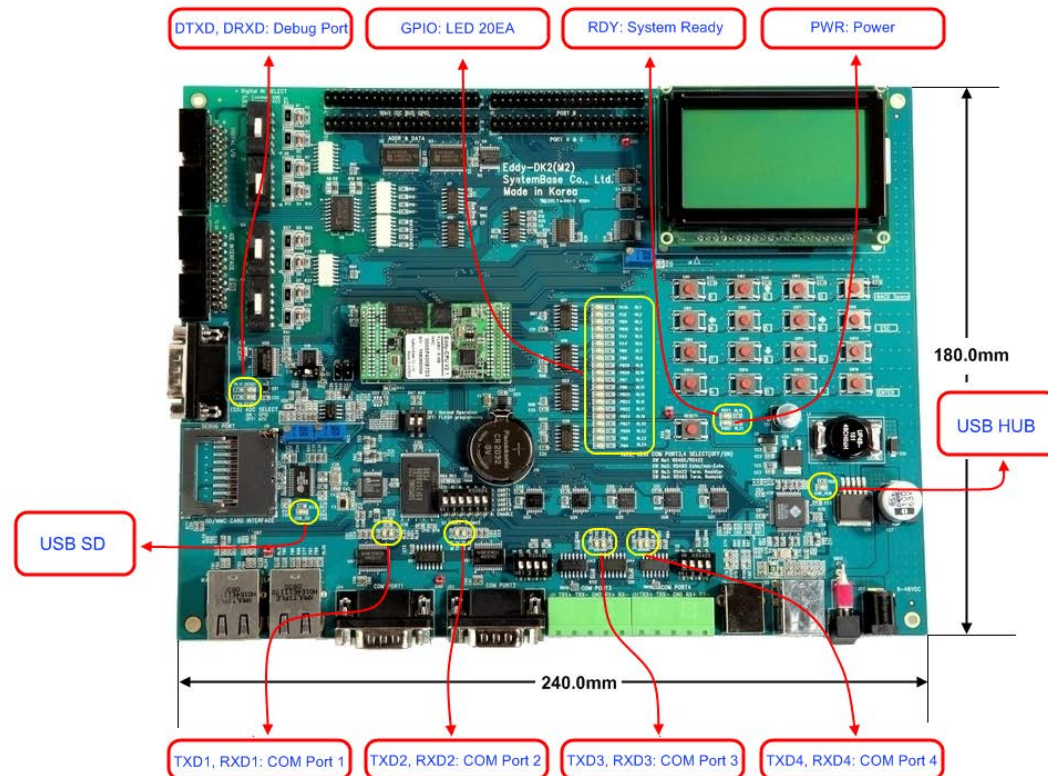
2.4.2.7. SW17: Power

전원을 인가한다.

2.4.2.8. Reset1:Reset

핀이름	기 능	용 도	I/O
PC16	nRESET	Reset key 입력 신호를 지속적으로 polling 하여 "Low" 지속시간을 check 하고 아래와 같이 동작한다. 5 초미만 : 일반적인 reset 기능 5 초이상 : Factory Default 기능	IN

2.4.3 LED 설명 (LED Description)



2.4.3.1. GPIO LED

제공되는 GPIO 포트는 최대 56개 이다. Eddy-DK 에서는 56개의 GPIO 중 Serial Port 와 연계된 GPIO 20개를 LED

를 통해서 확인 할 수 있다. 이때 GPIO와 Serial Port의 사용은 2.4.2.4.4 UART Select 를 통해서 제어 한다.

핀이름	기 능	용 도	I/O
PC10	CTS3	UART #3 Clear to Send	I
PC8	RTS3	UART #3 Request to Send	O
PB11	RXD3	UART #3 Receive Data	I
PB10	TXD3	UART #3 Transmit Data	O
PA5	CTS2	UART #2 Clear to Send	I
PA4	RTS2	UART #2 Request to Send	O
PB9	RXD2	UART #2 Receive Data	I
PB8	TXD2	UART #2 Transmit Data	O
PB29	CTS1	UART #1 Clear to Send	I
PB28	RTS1	UART #1 Request to Send	O
PB7	RXD1	UART #1 Receive Data	I
PB6	TXD1	UART #1 Transmit Data	O
PB25	RI0	UART #0 Ring Indicator	I
PB23	DCD0	UART #0 Data Carrier Detection	I
PB22	DSR	UART #0 Data Set Ready	O
PB24	DTR0	UART #0 Data Terminal Ready	I
PB27	CTS0	UART #0 Clear to Send	I
PB26	RTS0	UART #0 Request to Send	O
PB5	RXD0	UART #0 Receive Data	I
PB4	TXD0	UART #0 Transmit Data	O

참고로 PIO 라인은 high-drive current capable 이 있어서 PC4-PC31(2mA)을 제외한 PIO 라인은 16mA를 드라이브 할 수 있다. (CPU Datasheet의 41.2 DC characteristics, 아래표 참조)

41.2 DC Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Units
I _o	Output Current	PA0-PA31 PB0-PB31 PC0-PC3			16	mA
		PC4 - PC31 in 3.3V range			2*	
		PC4 - PC31 in 1.8V range			4	

* Eddy DK v2.1은 3.3V range 이므로 PC4-PC31 PIO는 2mA를 드라이브할 수 있다.

2.4.3.2. Power, Ready LED

System Ready (RDY): 시스템이 정상동작하고 있음을 알린다. (정상: 점멸 상태)

Power (PWR): 전압이 인가 되었음을 의미한다. (RED LED ON 상태)

2.4.3.3. Debug Port LED

DTXD (Debug Port Transmit Data LED) : Debug Port의 송신 상태를 알린다.

DRXD (Debug Port Receive Data LED) : Debug Port의 수신 상태를 알린다.

2.4.3.4. COM Port 1 LED

COM Port 1 Transmit LED : COM Port 1 의 송신 상태를 알린다.

COM Port 1 Receive LED : COM Port 1 의 수신 상태를 알린다.

2.4.3.5. COM Port 2 LED

COM Port 2 Transmit LED : COM Port 2의 송신 상태를 알린다.

COM Port 2 Receive LED : COM Port 2의 수신 상태를 알린다.

2.4.3.6. COM Port 3 LED

COM Port 3 Transmit LED : COM Port 3 의 송신 상태를 알린다.

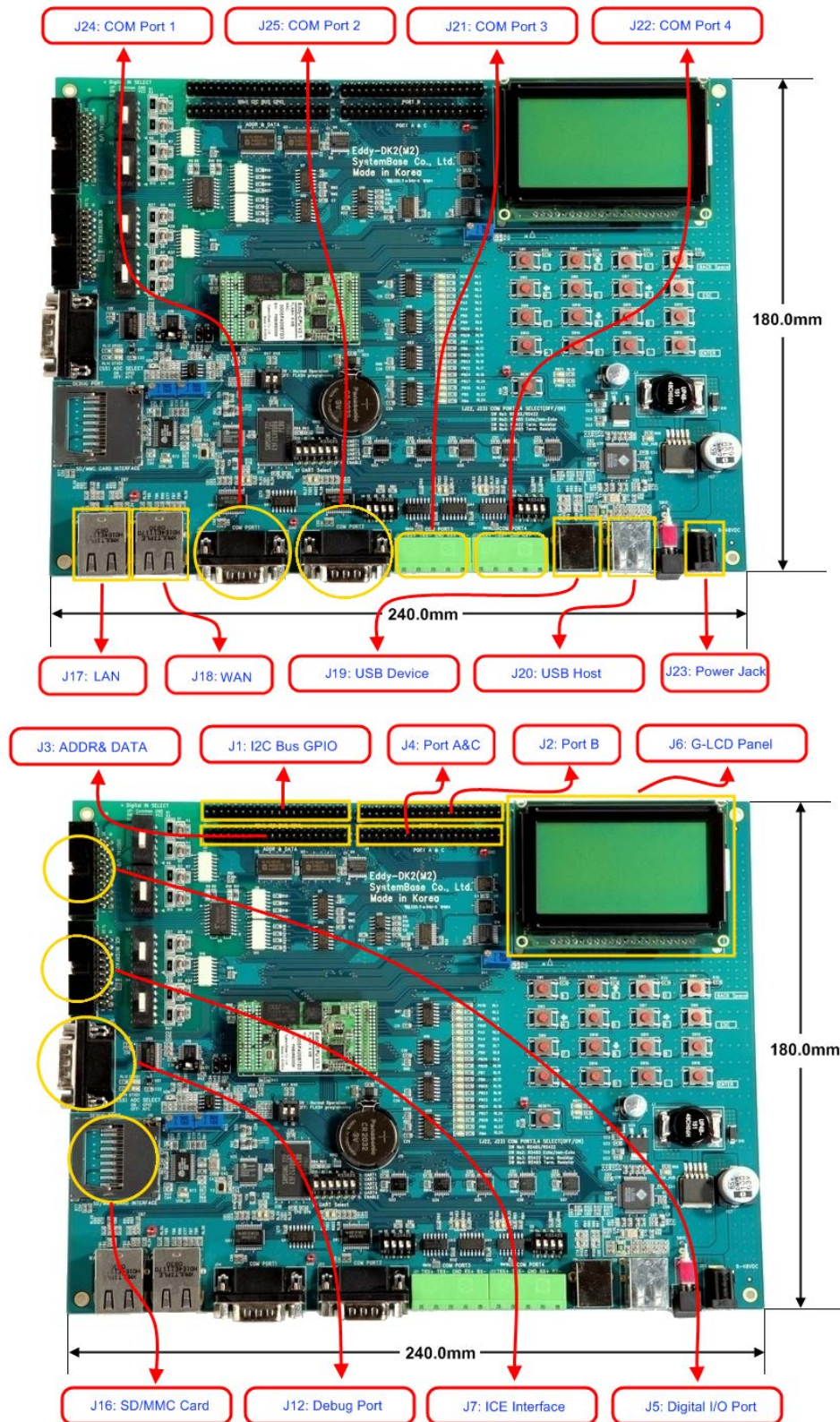
COM Port 3 Receive LED : COM Port 3 의 수신 상태를 알린다.

2.4.3.7. COM Port 4 LED

COM Port 4 Transmit LED : COM4 Port 4 의 송신 상태를 알린다.

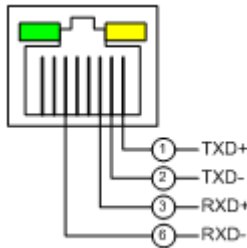
COM Port 4 Receive LED : COM4 Port 4 의 수신 상태를 알린다.

2.4.4 외부 디바이스 연결 설명(External Device Interface Description)



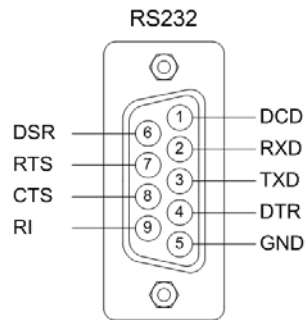
2.4.4.1. WAN & LAN Interface

LAN Port 는 자동으로 Cross/Direct를 인식한다. (Auto MDI/MDIX)



Pin	Signal	Description
1	TXD+	Transmit Data +
2	TXD-	Transmit Data -
3	RXD+	Receive Data +
6	RXD-	Receive Data -
LED	Description	
Left Green	100BaseT Link 시 ON 되며, 10BaseT Link 시 Off 된다.	
Right Yellow	네트워크 접속 시 On 상태이며, 데이터 송수신 시 점멸된다.	

2.4.4.2. COM Port 1 & COM Port 2

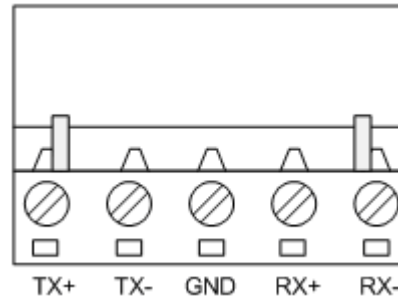


DB9 Male (COM Port 1, 2 공통)

RS232

Pin	Signal	Description
1	DCD	Data Carrier Detection (Input) (COM Port 1 only)
2	RXD	Receive Data (Input)
3	TXD	Transmit Data (Output)
4	DTR	Data Terminal Ready (Output) (COM Port 1 only)
5	GND	Ground
6	DSR	Data Set Ready (input) (COM Port 1 only)
7	RTS	Request to Send (Output)
8	CTS	Clear to Send (Input)
9	RI	Ring Indicator (Input)

2.4.4.3. COM Port 3 & COM Port 4



RS422 Full Duplex

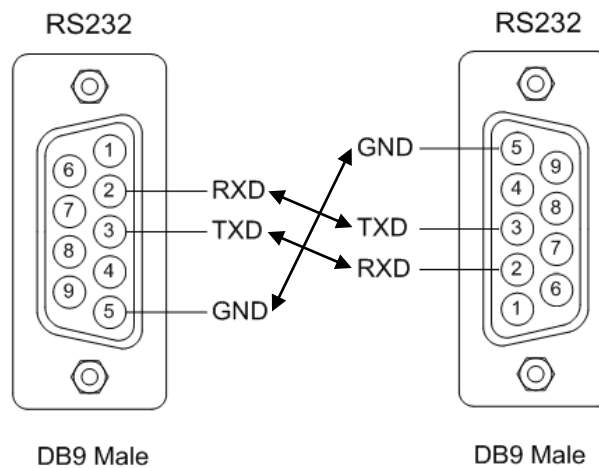
Pin	Signal	Description
1	TXD+	Transmit differential data positive (Output)
2	TXD-	Transmit differential data negative (Output)
3	GND	Ground
4	RXD+	Receive differential data positive (Input)
5	RXD-	Receive differential data negative (input)

RS485 Half Duplex

Pin	Signal	Description
1	TRX+	Transmit/Receive differential data positive
2	TRX-	Transmit/Receive differential data negative

2.4.4.4. Debug Port

디버그 포트를 통해서 제품의 디버그 메시지나 상태 정보를 확인 할 수 있다.



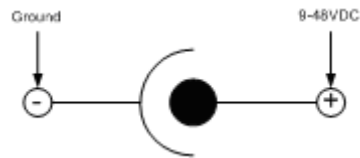
환경 설정

디버그 포트는 다음과 같이 환경이 설정이 되어 있으므로 사용자는 디버그 포트에 연결된 PC의 시리얼 포트에 설정을 다음과 같이 한다.

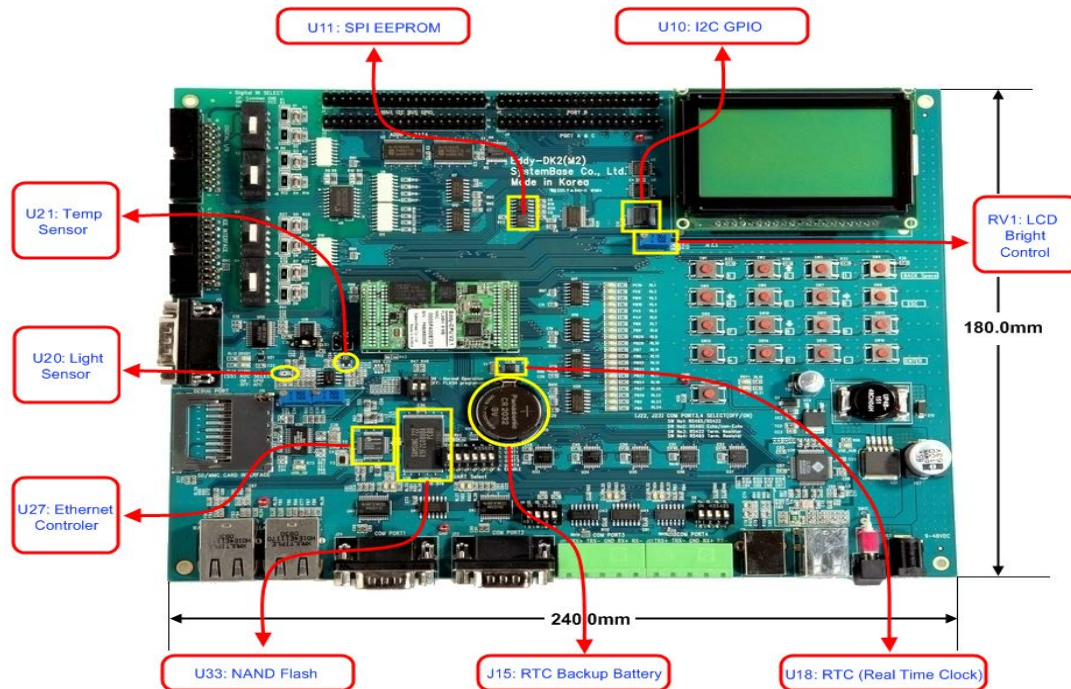
- Speed : 115200 bps
- Data bit : 8 bit
- Parity bit : Non Parity
- Stop bit : 1 bit
- Flow control : none

2.4.4.5. Power Jack

Contact	Polarity
Center (D : 2mm)	9-48VDC
Outer (D: 6.5mm)	Ground



2.4.5 내부 디바이스 설명(Internal Device Description)



2.4.5.1. EEPROM

Eddy DK v2.1 에는 SPI1에 하나의 EEPROM이 장착되어 있다.
SPI1 : AT25160, 2Kx8bit

2.4.5.2. LCD Module

Graphic LCD Module인 PowerTIP PG12864LRU-JCNH11Q 와 I2C-Bus I/O Expander IC PCA9539로 연결되어 있다.

Signal Name	기 능	용 도	I/O
P[00:07]	Data bits	Used for data transfer between the CPU and the LCD module.	I/O
P10	/CS1	Chip enable for D2 (Segment 1 to 64)	IN
P11	/CS2	Chip enable for D3 (Segment 65 to 128)	IN
P12	R/W	R/W signal input is used to select the read /write mode High = Read mode, Low = Write mode	IN
P13	D/ \overline{I}	Register selection input High = Data register Low = Instruction register (for write) Busy flag address counter (for read)	IN
P14	E	Start enable signal to read or write the data	IN

2.4.5.3. 16bit I2C Bus GPIO

I2C 인터페이스에 연결하여 16-bit I/O 확장이 가능한 PCA9539를 사용하였다. Eddy DK v2.1에는 Slave address가 0x74로 설정되어 있고 이것은 A1,A0 address input 설정에 따라 0x74에서 0x77까지 변경이 가능하다.

16-bit I/O는 아래 표와 같이 Digital Input/Output으로 사용되고, Expansion Header와도 연결이 되어있어 GPIO로 사용이 가능하다. GPIO로 사용시는 개별적으로 입출력 설정이 가능하다.

기 능	용 도	I/O
P00-P07	DIO Output, 즉 DO[0:7] 에 순차적으로 연결 된다.	OUT
P00	DIO output, DO0 제어	
P01	DIO output, DO1 제어	
P02	DIO output, DO2 제어	
P03	DIO output, DO3 제어	
P04	DIO output, DO4 제어	
P05	DIO output, DO5 제어	
P06	DIO output, DO6 제어	
P07	DIO output, DO7 제어	
P10-P17	DIO Input, 즉 DI[0:7] 에 순차적으로 연결 된다.	IN
P10	DIO Input, DI0 입력	
P11	DIO Input, DI1 입력	
P12	DIO Input, DI2 입력	
P13	DIO Input, DI3 입력	
P14	DIO Input, DI4 입력	
P15	DIO Input, DI5 입력	
P16	DIO Input, DI6 입력	
P17	DIO Input, DI7 입력	IN
/INT	Eddy-CPU PB16 에 연결	OUT

2.4.5.4. RTC

- I2C 인터페이스에 연결된 DS1340을 사용하였다.
- DS1340은 load capacitance = 12.5pF 인 crystal 을 사용해야 한다. (아래 Crystal Spec 참조)
- RTC Chip에 따라 Crystal 규격이 다르므로 확인 후 부품을 선정해야 한다.
- Backup Battery로는 CR2032(235mAh) Lithium 배터리를 사용하였다.

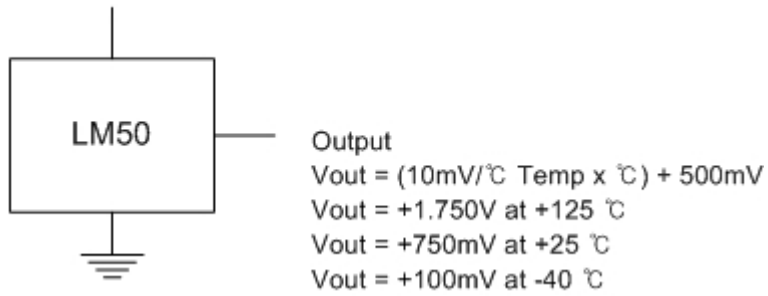
DS1340 Crystal Specifications

Parameter	Symbol	MIN	TYP	MAX	Units
Normal Frequency	fo		32.768		KHz
Series Resistance	ESR			45,60	K Ω
Load Capacitance	CL		12.5		pF

2.4.5.5. Temp Sensor

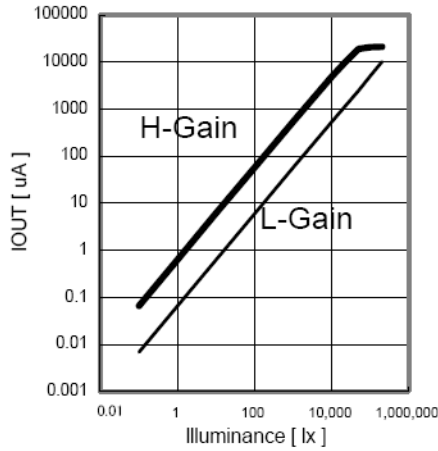
AD0(PC0)에 National 의 LM50 을 사용하였다.

+Vs (4.5V to 10V)



2.4.5.6. Light Sensor

Rohm의 BH1600FVC를 사용하였고 조도 대비 출력전류는 아래와 같다.



The Output voltage is calculated as below

$$V_{iout} = 0.6 \times 10^{-6} \times E_v \times R_1$$

Where, V_{iout} = IOUT output voltage [V]

E_v = lilluminance of the ALS(Ambient Light Sensor) surface [lx]

R_1 = IOUT output resistor [Ω]

2.4.5.7. NAND Flash

- 256MB, 8bit 메모리 (Samsung K9F2G08U0A-PCB0)

- Chip Select #3 사용, Address 영역 : 0x4000_0000~0x4FFF_FFFF.

Eddy-CPU v2.1/V2.5 Signal Name	기 능	용 도	I/O
A22	CLE	COMMAND LATCH ENABLE The CLE input controls the activating path for commands sent to the command register.	OUT
A21	ALE	ADDRESS LATCH ENABLE The ALE input controls the activating path for address to the internal address registers.	OUT
NANDOE	NANDOE	data-out control	OUT
NANDWE	NANDWE	controls writes to the I/O port	OUT
PC14(NCS3)	NANDCS	device selection control	OUT
PC17	RDYBSY (R/B)	READY/BUSY OUTPUT The R/B output indicates the status of the device operation. When low, it indicates that a program, erase or random read operation is in process and returns to high state upon completion. It is an open drain output and does not float to high-z condition when the chip is deselected or when outputs are disabled.	IN

D[0:7]	DATA bits	DATA INPUTS/OUTPUTS The I/O pins are used to input command, address and data, and to output data during read operations. The I/O pins float to high-z when the chip is deselected or when the outputs are disabled.	I/O
--------	-----------	---	-----

2.4.5.8. Ethernet Controller (WAN Port)

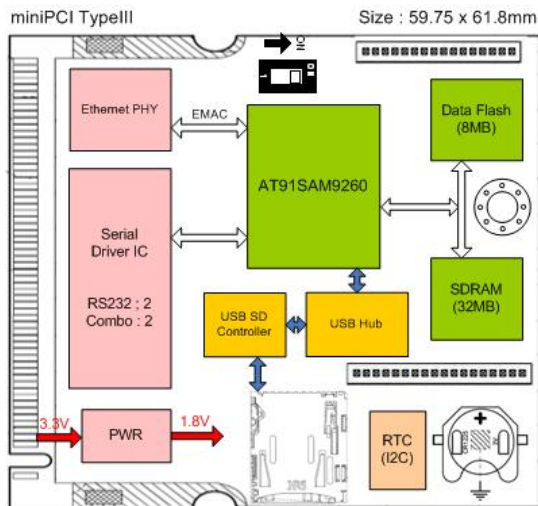
- Davicom DM9000B Ethernet Controller를 사용하여 16bit mode로 연결되어 있다..
- 내부 pull-down 되어있는 Strap option pin 인 EECS pin에 외부 pull-up 저항을 연결해야 LED가 동작한다.
- RJ45 Transformer Center Tap에 연결되는 전원은 반드시 DM9000B AVDD18와 연결해야 한다.

Eddy-CPU v2.1/V2.5(B) Signal Name	DM9000B Signal Name	Description	I/O
PC12/NCS7	CSN	Chip Select #7 Address : 0x8000 0000-0x8FFF FFFF	OUT
PC15/IRQ1	INTRN	인터럽트 극성은 EECK(pin20) 설정에 따른다. 1 : INT pin low active 0 : INT pin high active DK v2.1 에서는 EECK 가 float 되어 active high 로 동작	IN
A2	CMD	Command Type When high, Data port When low, INDEX port	OUT
D[0:15]	Data Bus	16-bit mode 연결	I/O

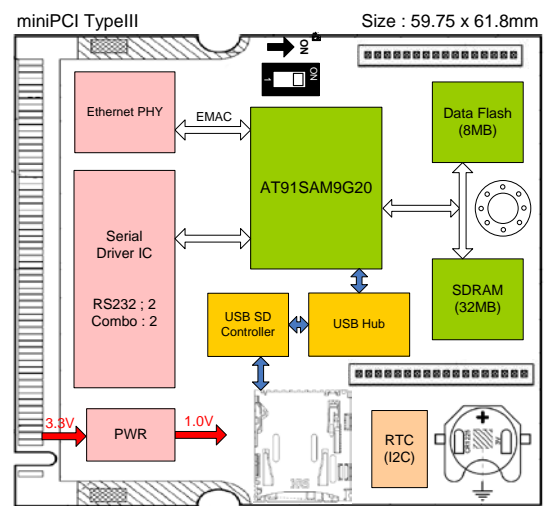
2.5 Eddy-S4M v2.1 / v2.5 / v2.5B

ARM9 프로세서와 32MB SDRAM, 8MB DataFlash, 10/100Base-T Ethernet 포트, 최대 34 ea 의 사용자 Programmable IO 와 MicroSD, RTC, Backup 배터리, 4 포트 시리얼(2*RS232, 2*Combo) 를 갖춘 miniPCI 타입의 Embedded module 이다.

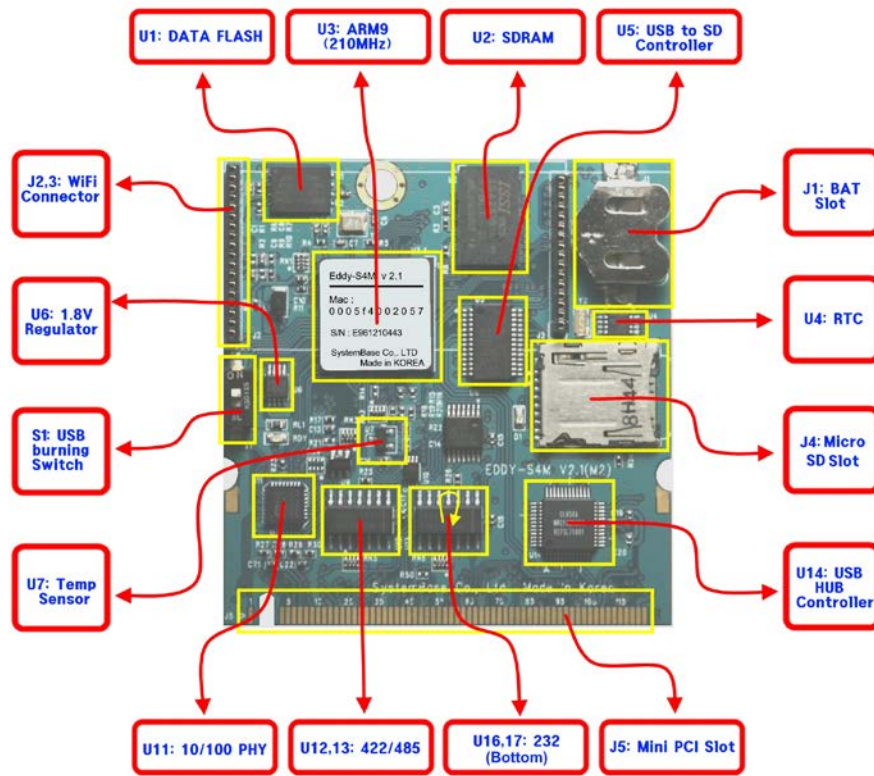
Eddy-S4M 크기는 59.75 x 61.8mm 로, 별도로 제공하는 Eddy-S4M-JIG 보드를 사용하는 경우, 사용자는 별도의 하드웨어 개발 없이도 응용제품을 쉽게 개발 가능함으로써 개발에 따른 시간과 비용을 최소화 되도록 하였다. 또한 제공되는 예제 소스 코드와 Evaluation Kit 회로를 참조하여 사용자가 원하는 디바이스를 간편하게 구현할 수 있다. 그리고, Eddy-S4M v2.1과 Eddy-S4M v2.5는 호환되며 같은 DK보드와 JIG보드를 사용한다.



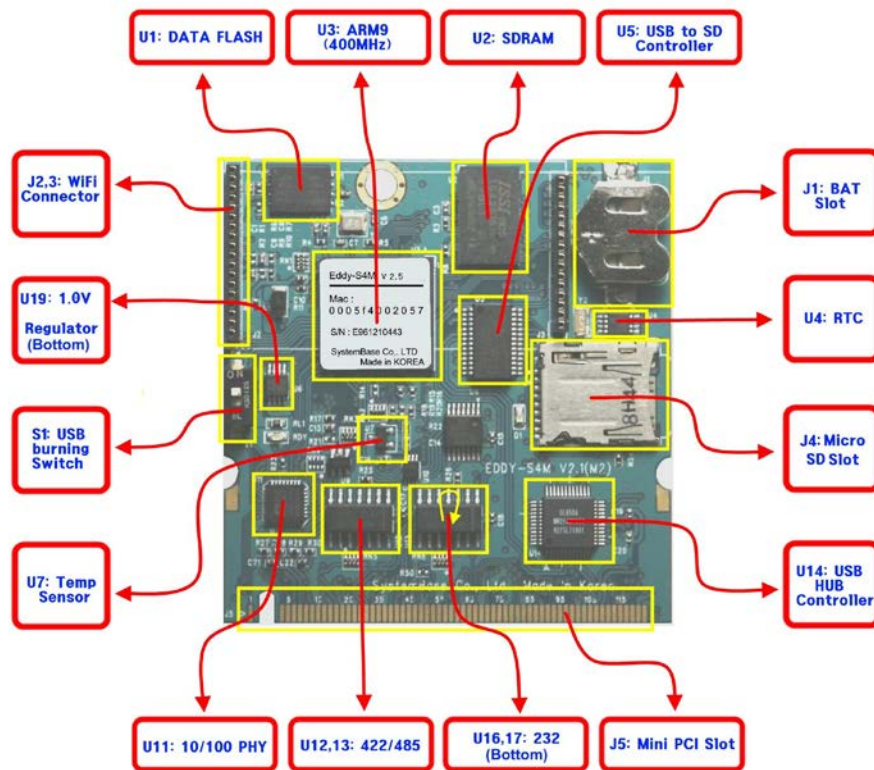
[Eddy-S4M v2.1 Block Diagram]



[Eddy-S4M V2.5 Block Diagram]



[Eddy-S4M v2.1]



[Eddy-S4M v2.5]

2.5.1 miniPCI Card Type III Connector Pinout (J5)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	JTAG_TDI	2	JTAG_TDO	63	3.3V	64	PB13
	Key		Key	65	PB16	66	PB17
3	JTAG_TMS	4	JTAG_RTCK	67	PB18	68	PB19
5	JTAG_TCK	6	ICE_NTRST	69	GND	70	3.3V
7	LAN_RX+	8	LAN_TX+	71	PB20	72	PB21
9	LAN_RX-	10	LAN_TX-	73	PB30	74	GND
11	LAN_Speed	12	LAN_LINK	75	PC0	76	PB31
13	P3_RX-	14	RDY#	77	GND	78	PC1
15	GND	16	NC	79	PC2	80	PC3
17	P3_RX+	18	NC	81	PC5	82	GND
19	3.3V	20	DCD0	83	GND	84	PC9
21	P3_TX+	22	DTR0	85	PC10	86	PC12
23	GND	24	3.3V	87	PC13	88	3.3V
25	P3_TX-	26	nRESET	89	3.3V	90	PC14
27	GND	28	3.3V	91	PC15	92	PC17
29	P4_RX+	30	RxD0#	93	PC18	94	PC19
31	3.3V	32	GND	95	PC24	96	PC20
33	P4_RX-	34	RTS0	97	NC	98	PC25
35	P4_TX+	36	TxD0#	99	I2C_TWCK	100	I2C_TWD
37	GND	38	CTS0	101	GND	102	GND
39	P4_TX-	40	3.3V	103	DDM	104	DDP
41	DEBUG_TxD	42	DSR0	105	DM2	106	DP2
43	DEBUG_RxD	44	RI0	107	DM3	108	DP3
45	PA5	46	RxD1#	109	DM4	110	DP4
47	PA22	48	RTS1	111	SDDATA0	112	SDDATA1
49	GND	50	GND	113	SDDATA2	114	GND
51	PA30	52	TxD1#	115	SDCMD	116	SDDATA3
53	NC	54	CTS1	117	SDCDN	118	SDCLK
55	GND	56	NRST	119	JTAG_SEL	120	SDWP
57	PB0	58	PB1	121	NC	122	BMS
59	PB2	60	PB3	123	NC	124	3.3V
61	PB12	62	GND				

2.5.2 보드별 Connector Pinout

2.5.2.1. ICE and JTAG

S4M Pin No (124)	Name	S4M-JIG Pin HDR (46*2)	S4M-DK Pin HDR (46*2)	Description
1	TDI	-	-	Test Data IN
2	TDO	-	-	Test Data Out
3	TMS	-	-	Test Mode Select
4	RTCK	-	-	Return Test Clock
5	TCK	-	-	Test Clock
6	NTRST	-	-	Test Reset
119	JTAGSEL	-	-	JTAG boundary scan can be used by connecting J3. This pin should not be connected when using ICE (In-Circuit Emulator) or in normal operation status.

2.5.2.2. Ethernet signal from or to PHYceiver

S4M Pin No (124)	Name	S4M-JIG Pin HDR (46*2)	S4M-DK Pin HDR (46*2)	Description		
7	LAN_RX+	J5 pin2	J7 pin2	CPU 내부 Ethernet PHY Physical receive or transmit signal (+ differential)		
8	LAN_TX+	J5 pin1	J7 pin1	CPU 내부 Ethernet PHY Physical receive or transmit signal (- differential)		
9	LAN_RX-	J5 pin3	J7 pin3	CPU 내부 Ethernet PHY Physical receive or transmit signal (+ differential)		
10	LAN_TX-	J5 pin4	J7 pin4	CPU 내부 Ethernet PHY Physical receive or transmit signal (- differential)		
11	LAN_Speed	J5 pin6	J7 pin6	LAN connection status LED		
				Link/Activity	Pin State	LED Definition
				No Link	H	OFF
				Link	L	ON
Activity	Toggle	Blinking				
12	LAN_Link	J5 pin5	J7 pin5	Link/Activity	Pin State	LED Definition
				No Link	H	OFF
				Link	L	ON
				Activity	Toggle	Blinking

2.5.2.3. Serial (RS232 & COMBO) and PIOA (Peripheral I/O Controller A)

S4M Pin No (124)	Name	S4M-JIG Pin HDR (46*2)	S4M-DK Pin HDR (46*2)	Description
13	P2_RX-	J4 pin20	J6 pin20	COM port #3 Receive differential data negative (Input) Eddy-S4M 모듈내 RS422/485 inverting receiver input
14	RDY#	J4 pin45	J6 pin45	CPU의 동작 상태 표시 (정상: 점멸)
17	P2_RX+	J4 pin19	J6 pin19	COM port #3 Receive differential data positive (Input) Eddy-S4M 모듈내 RS422/485 Noninverting receiver input
20	DCD0	J4 pin9	J6 pin9	COM port #1 Data Carrier Detection signal Eddy-S4M 모듈내 RS232 receiver input
21	P2_TX+	J4 pin17	J6 pin17	COM port #3 Transmit differential data positive (Output) Eddy-S4M 모듈내 RS422/485 Noninverting driver output
22	DTR0	J4 pin7	J6 pin7	COM port #1 Data Terminal Ready signal Eddy-S4M 모듈내 RS232 driver output
25	P2_TX-	J4 pin18	J6 pin18	COM port #3 Transmit differential data negative (Output) Eddy-S4M 모듈내 RS422/485 inverting driver output
26	nRESET	J4 pin46	J6 pin46	Reset Input. 외부 Reset key로 부터 입력 신호를 지속적으로 polling 하여, "Low"의 지속시간을 check 하여 S/W 적으로 아래와 같이 동작한다. 5 초미만 : 일반적인 reset 기능 5 초이상 : Factory Default 기능
29	P3_RX+	J4 pin23	J6 pin23	COM port #4 Receive differential data negative (Input) Eddy-S4M 모듈내 RS422/485 Noninverting receiver input
30	RxD0#	J4 pin4	J6 pin4	COM port #1 Receive Data signal Eddy-S4M 모듈내 RS232 receiver input
33	P3_RX-	J4 pin24	J6 pin24	COM port #4 Receive differential data negative (Input) Eddy-S4M 모듈내 RS422/485 inverting receiver input
34	RTS0	J4 pin5	J6 pin5	COM port #1 Request To Send signal Eddy-S4M 모듈내 RS232 driver output
35	P3_TX+	J4 pin21	J6 pin21	COM port #4 Transmit differential data positive (Output) Eddy-S4M 모듈내 RS422/485 Noninverting driver output
36	TxD0#	J4 pin3	J6 pin3	COM port #1 Transmit Data signal Eddy-S4M 모듈내 RS232 driver output
38	CTS0	J4 pin6	J6 pin6	COM port #1 Request to Send signal Eddy-S4M 모듈내 RS232 receiver input
39	P3_TX-	J4 pin22	J6 pin22	COM port #4 Transmit differential data negative(Output) Eddy-S4M 모듈내 RS422/485 inverting driver output
41	DTxD#	J4 pin1	J6 pin1	Transmit Data signal of Debug Port Eddy-S4M 모듈내 RS232 driver output
42	DSR0	J4 pin8	J6 pin8	COM port #1 Data Set Ready signal Eddy-S4M 모듈내 RS232 receiver input
43	DRxD	J4 pin2	J6 pin2	Receive Data signal of Debug Port Eddy-S4M 모듈내 RS232 receiver input
44	RI0	J4 pin8	J6 pin8	COM port #1 Ring Indicator signal Eddy-S4M 모듈내 RS232 receiver input
45	PA5	J5 pin7	J7 pin7	GPIO로만 사용가능
46	RxD1#	J4 pin12	J6 pin12	COM port #1 Receive Data signal Eddy-S4M 모듈내 RS232 receiver input
47	PA22	J5 pin8	J7 pin8	GPIO로 사용가능
48	RTS1	J4 pin13	J6 pin13	COM port #1 Request to Send signal

				Eddy-S4M 모듈내 RS232 driver output
51	PA30	J5 pin9	J7 pin9	GPIO로만 사용가능
52	TxD1#	J4 pin11	J6 pin11	COM port #1 Request to Send signal Eddy-S4M 모듈내 RS232 driver output
54	CTS1	J4 pin14	J6 pin14	COM port #1 Request to Send signal Eddy-S4M 모듈내 RS232 receiver input
56	NRST	J5 pin46	J7 pin46	External device Reset output signal (active low)

2.5.2.4. PIOB and PIOC (Peripheral I/O Controller B/C)

S4M Pin No (124)	Name	S4M-JIG Pin HDR (46*2)	S4M-DK Pin HDR (46*2)	Description	
57	PB0	J5 pin11	J7 pin11	Peripheral A : SPI1_MISO	SPI1(Serial Peripheral Interface) Master In Slave Out
				Peripheral B : TIOA3	Timer Counter ch3 I/O Line A
58	PB1	J5 pin12	J7 pin12	Peripheral A : SPI1_MOSI	SPI1(Serial Peripheral Interface) Master Out Slave In
				Peripheral B : TIOB3	Timer Counter ch3 I/O Line B
59	PB2	J5 pin13	J7 pin13	Peripheral A : SPI1_SPCK	SPI1(Serial Peripheral Interface) Serial Clock
60	PB3	J5 pin14	J7 pin14	Peripheral A : SPI1_NPCS0	SPI1(Serial Peripheral Interface) Peripheral Chip Select 0
				Peripheral B : TIOA5	Timer Counter ch5 I/O Line A
61	PB12	J5 pin17	J7 pin17	Peripheral A : TXD5	USART5 Transmit Data
64	PB13	J5 pin18	J7 pin18	Peripheral A : RXD5	USART5 Receive Data
65	PB16	J5 pin119	J7 pin119	Peripheral A : TK0	SSC Transmit Clock
				Peripheral B : TCLK3	Timer Counter ch3 External CLK IN
66	PB17	J5 pin20	J7 pin20	Peripheral A : TF0	SSC Transmit Frame Sync
				Peripheral B : TCLK4	Timer Counter ch4 External CLK IN
67	PB18	J5 pin21	J7 pin21	Peripheral A : TD0	SSC Transmit Data
				Peripheral B : TIOB4	Timer Counter ch4 I/O Line B
68	PB19	J5 pin22	J7 pin22	Peripheral A : RD0	SSC Receive Data
				Peripheral B : TIOB5	Timer Counter ch5 I/O Line B

71	PB20	J5 pin23	J7 pin23	Peripheral A : RK0	SSC Receive Clock
72	PB21	J5 pin24	J7 pin24	Peripheral A : RF0	SSC Receive Frame Sync
73	PB30	J5 pin25	J7 pin25	Peripheral A : PCK0	Programmable Clock Output 0
75	PC0	J5 pin27	J7 pin27	Peripheral A : AD0	Analog to Digital Converter Input Ch0
76	PB31	J5 pin26	J7 pin26	Peripheral A : PCK1	Programmable Clock Output 1
78	PC1	J5 pin28	J7 pin28	Peripheral A : AD1	Analog to Digital Converter Input Ch1
				Peripheral B : PCK0	Programmable Clock Output 0
79	PC2	J5 pin29	J7 pin29	Peripheral A : AD2	Analog to Digital Converter Input Ch2
				Peripheral B : PCK1	Programmable Clock Output 1
80	PC3	J5 pin30	J7 pin30	Peripheral A : AD3	Analog to Digital Converter Input Ch3
				Peripheral B : SPI1_NPCS3	SPI1(Serial Peripheral Interface) Peripheral Chip Select 3
81	PC5	J5 pin33	J7 pin33	Peripheral B : SPI1_NPCS1	SPI1(Serial Peripheral Interface) Peripheral Chip Select 1
84	PC9	J5 pin34	J7 pin34	GPIO로만 사용가능	
85	PC10	J5 pin35	J7 pin35	GPIO로만 사용가능	
86	PC12	J5 pin36	J7 pin36	GPIO로만 사용가능	
87	PC13	J5 pin37	J7 pin37	GPIO로만 사용가능	
90	PC14	J5 pin38	J7 pin38	GPIO로만 사용가능	
91	PC15	J5 pin39	J7 pin39	GPIO로만 사용가능	
92	PC17	J5 pin40	J7 pin40	GPIO로만 사용가능	
93	PC18	J5 pin41	J7 pin41	Peripheral B : SPI1_NPCS1	SPI1(Serial Peripheral Interface) Peripheral Chip Select 1
94	PC19	J5 pin42	J7 pin42	Peripheral B : SPI1_NPCS2	SPI1(Serial Peripheral Interface) Peripheral Chip Select 2
95	PC24	J5 pin44	J7 pin44	GPIO로만 사용가능	
96	PC20	J5 pin43	J7 pin43	Peripheral B : SPI1_NPCS3	SPI1(Serial Peripheral Interface) Peripheral Chip Select 3
98	PC25	J5 pin45	J7 pin45	GPIO로만 사용가능	

2.5.2.5. Two Wire Interface

S4M Pin No (124)	Name	S4M-JIG Pin HDR (46*2)	S4M-DK Pin HDR (46*2)	Description
99	I2C_TWCK	J4 pin43	J6 pin43	Two-wire Serial Clock. RTC 기능을 사용하지 않을 경우, 이 핀은 GPIO로 사용 가능하다.
100	I2C_TWD	J4 pin44	J6 pin44	Two-wire Serial Data. RTC 기능을 사용하지 않을 경우, 이 핀은 GPIO로 사용 가능하다.

2.5.2.6. Universal Serial Bus

S4M Pin No (124)	Name	S4M-JIG Pin HDR (46*2)	S4M-DK Pin HDR (46*2)	Description
103	DDM	J4 pin25	J6 pin25	USB Device Port Data -
104	DDP	J4 pin26	J6 pin26	USB Device Port Data +
105	DM2	J4 pin27	J6 pin27	USB Port2 Data -, GL850A USB 2.0 Hub Controller의 DSPORT2 와 연결되어 있다.
106	DP2	J4 pin27	J6 pin27	USB Port2 Data +, GL850A USB 2.0 Hub Controller의 DSPORT2 와 연결되어 있다.
107	DM3	J4 pin29	J6 pin29	USB Port3 Data -, GL850A USB 2.0 Hub Controller의 DSPORT3 와 연결되어 있다.
108	DP3	J4 pin30	J6 pin30	USB Port3 Data +, GL850A USB 2.0 Hub Controller의 DSPORT3 와 연결되어 있다.
109	DM4	J4 pin33	J6 pin33	USB Port4 Data -, GL850A USB 2.0 Hub Controller의 DSPORT4 와 연결되어 있다.
110	DP4	J4 pin34	J6 pin34	USB Port4 Data +, GL850A USB 2.0 Hub Controller의 DSPORT4 와 연결되어 있다.

2.5.2.7. Multimedia Card Interface

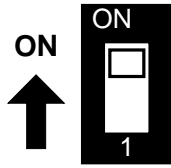
S4M Pin No (124)	Name	S4M-JIG Pin HDR (46*2)	S4M-DK Pin HDR (46*2)	Description
111	SDDATA0	J4 pin35	J6 pin35	SD Data0
112	SDDATA1	J4 pin36	J6 pin36	SD Data1
113	SDDATA2	J4 pin37	J6 pin37	SD Data2
115	SDCMD	J4 pin38	J6 pin38	SD command
116	SDDATA3	J4 pin39	J6 pin39	SD Data3
117	SDCDN	J4 pin40	J6 pin40	SD card detect
118	SDCLK	J4 pin41	J6 pin41	SD Clock
120	SDWP	J4 pin42	J6 pin42	SD Write Protect

122	BMS	-	-	Boot Mode Select signal BMS = 1, Boot on Embedded ROM BMS = 0, Boot on External Memory
-----	-----	---	---	--

2.5.2.8. etc

S4M Pin No (124)	Name	S4M-JIG Pin HDR (46*2)	S4M-DK Pin HDR (46*2)	Description
16, 18, 53, 97, 121, 123	NC	J5 pin10	J5 pin10	No Connection
15, 23, 27, 32, 37, 49, 50, 55, 62, 69, 74, 77, 82, 83, 101, 102, 114	GND	J4: 31,32 J5: 31,32	J6: 31,32 J7: 31,32	Ground
19, 24, 28, 31, 40, 63, 70, 88, 89, 124	3.3V	J4: 15,16	J6: 15,16	3.0 to 3.6V power input

2.5.3 스위치 동작



Switch No 1	Operation description
OFF	For Flash Programming USB Device 포트를 통해 펌웨어 이미지를 Flash 메모리에 저장하는 방식 (Windows Host 를 통해서만 가능) 으로 자세한 사용방법은 본 매뉴얼의 9 장 시스템 복구를 참조한다.
ON	Eddy-S4M v2.1 Data Flash 를 통해 정상적으로 부팅한다.

2.5.4 LED 동작

System Ready (RDY):로 시스템이 정상동작하고 있음을 알린다. (정상: 점멸 상태)

2.5.5 Ethernet

Eddy-S4M 모듈내 KSZ8041NL PHY가 내장되어 있기 때문에 트랜스 포머가 내장된 RJ45 커넥터만 연결하여 간단하게 Ethernet을 구현하였다.

WARNING : 트랜스포머가 내장된 RJ45 사용시 제품마다 내부회로가 다를 수 있다. 따라서 보드 설계시, 반드시 RJ45 커넥터 내부 회로의 핀번호를 확인하고 설계해야 한다.

KSZ8041NL의 기능을 아래와 같다.

- Fully compliant to IEEE 802.3u Standard
- Supports MDI/MDI-X auto crossover (Auto-MDI)
- MII interface support
- RMI interface support with external 50MHz system clock
- ESD rating (6kV)
- Built-in 1.8V regulator for core
- Available in 32-pin (5mm x 5mm) MLF® package

2.5.6 RTC

- I2C 인터페이스에 연결된 DS1340을 사용하였다.
- DS1340은 load capacitance = 12.5pF 인 crystal 을 사용해야 한다. (아래 Crystal Spec 참조)
- RTC Chip에 따라 Crystal 규격이 다르므로 확인 후 부품을 선정해야 한다.
- Backup Battery로는 CR2032(235mAh) Lithium 배터리를 사용하였다.

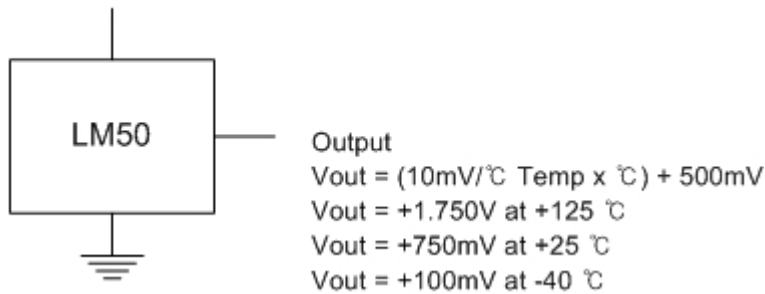
DS1340 Crystal Specifications

Parameter	Symbol	MIN	TYP	MAX	Units
Normal Frequency	fo		32.768		KHz
Series Resistance	ESR			45,60	K Ω
Load Capacitance	CL		12.5		pF

2.5.7 Temp Sensor

AD0(PC0)에 National 의 LM50 을 사용하였다.

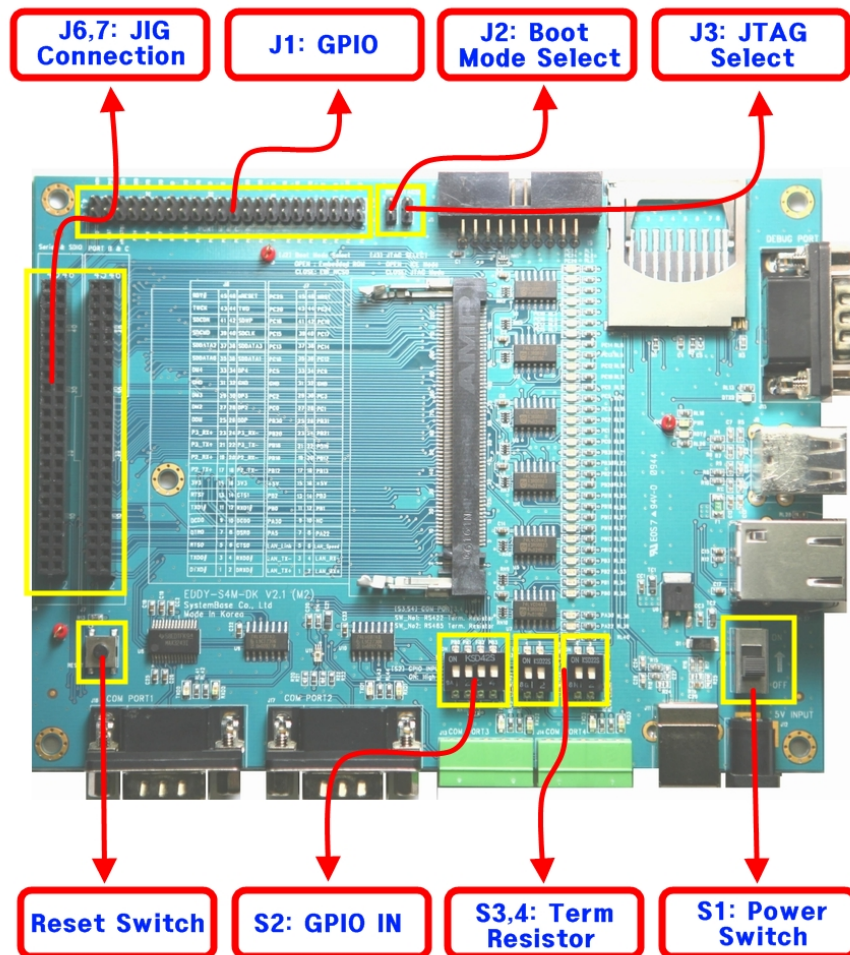
+Vs (4.5V to 10V)



2.6 Eddy-S4M-DK v2.1

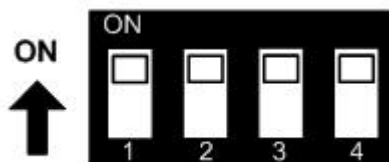
Eddy-S4M 개발용 키트인 DK 는 Eddy-S4M 을 탑재하여 프로그래머가 쉽게 자신의 어플리케이션을 탑재하고 테스트할 수 있도록 도움을 주는 개발 키트이다.

2.6.1 스위치 및 커넥터 설명



2.6.1.1. S2 : GPIO 입력 설정

PB0-PB4를 입력으로 설정하고 스위치를 조작하여 입력값이 바뀌는지 확인할 수 있다.



Switch No		Down Position(OFF)	UP Position(ON)
1	PB0 입력 값	Low	High
2	PB0 입력 값	Low	High
3	PB0 입력 값	Low	High
4	PB0 입력 값	Low	High

2.6.1.2. S3,4 : 종단 저항 선택

COM Port #3 과 COM Port #4는 RS422/RS485를 지원하는 Combo 포트이다. 이들 포트에서 종단저항은 각 포트의 Terminal Block 위에 위치한 스위치를 통해서 설정된다.

오류! 편집 중 필드 코드에서는 개체를 만들 수 없습니다.

Switch No	Down Position(OFF)	UP Position(ON)
1	RS422 Termination Resistor not connected	RS422 Termination Resistor Connected
2	RS485 Termination Resistor not connected	RS422 Termination Resistor Connected
1	RS422 Termination Resistor not connected	RS422 Termination Resistor Connected
2	RS485 Termination Resistor not connected	RS422 Termination Resistor Connected

2.6.1.3. J6,J7 : JIG 보드 연결 connector(Socket)

J6

Pin	Signal	Pin	Signal
1	DTxD	2	DRxD
3	TxD0#	4	RxD0#
5	RTS0	6	CTS0
7	DTR0	8	DSR0
9	DCD0	10	RI0
11	TxD1#	12	RxD1#
13	RTS1	14	CTS1
15	3,3V	16	3,3V
17	P3_TX+	18	P3_TX-
19	P3_RX+	20	P3_RX-
21	P4_TX+	22	P4_TX-
23	P4_RX+	24	P4_RX-
25	DDM	26	DDP
27	DM2	28	DP2
29	DM3	30	DP3
31	GND	32	GND
33	DM4	34	DP4
35	SDDATA0	36	SDDATA1

J7

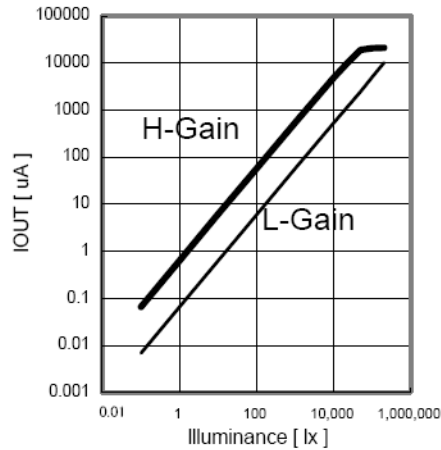
Pin	Signal	Pin	Signal
1	LAN_TX+	2	LAN_RX+
3	LAN_TX -	4	LAN_RX-
5	LAN_LINK	6	LAN_Speed
7	PA5	8	PA22
9	PA30	10	NC
11	PB0	12	PB1
13	PB2	14	PB3
15	5V	16	5V
17	PB12	18	PB13
19	PB16	20	PB17
21	PB18	22	PB19
23	PB20	24	PB21
25	PB30	26	PB31
27	PC0	28	PC1
29	PC2	30	PC3
31	GND	32	GND
33	PC5	34	PC9
35	PC10	36	PC12

37	SDDATA2	38	SDDATA3
39	SDCMD	40	SDCLK
41	SDCDN	42	SDWP
43	TWCK	44	TWD
45	RDY#	46	nRESET(IN)

37	PC13	38	PC14
39	PC15	40	PC17
41	PC18	42	PC19
43	PC20	44	PC24
45	PC25	46	NRST(OUT)

2.6.1.4. U7 : Light Sensor

Rohm의 BH1600FVC를 사용하였고 조도 대비 출력전류는 아래와 같다.



The Output voltage is caculated as below

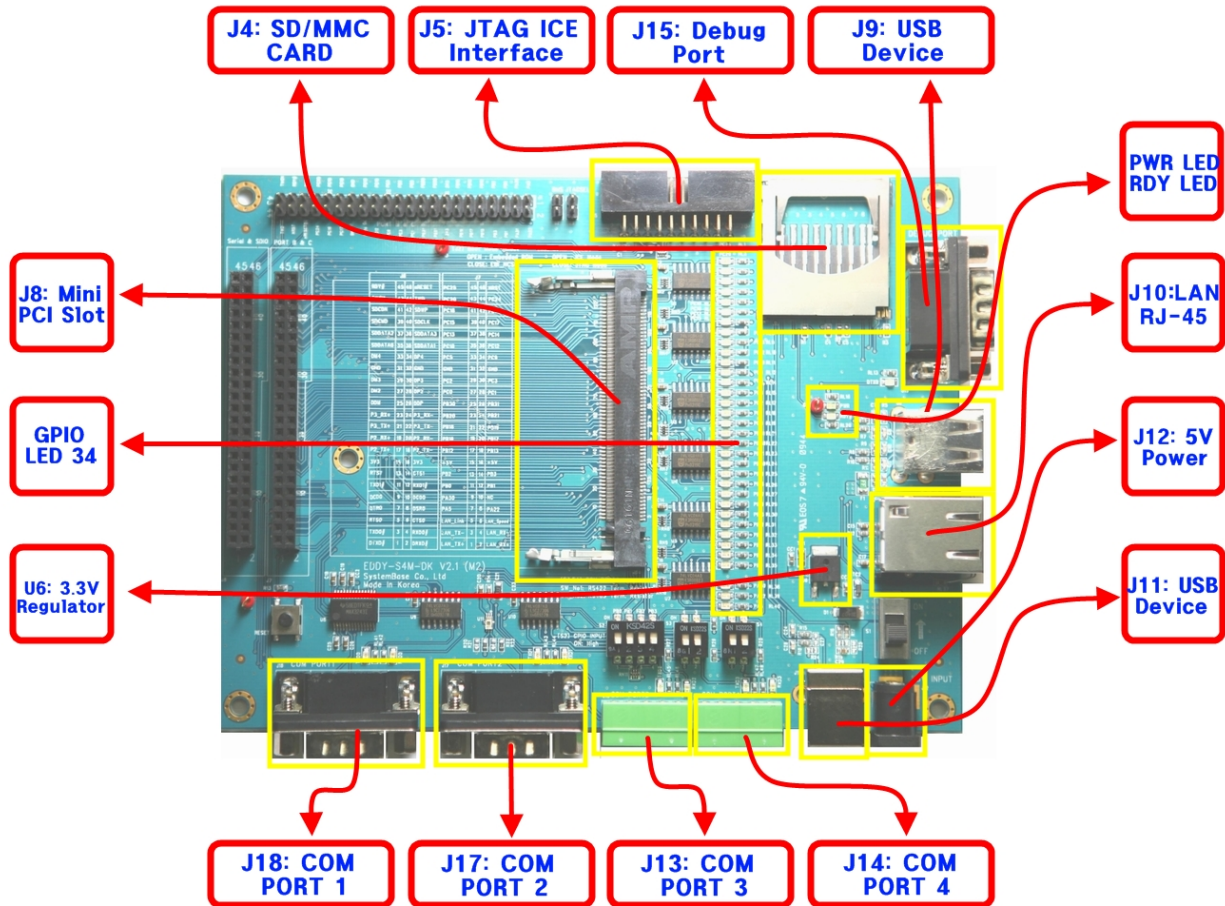
$$V_{iout} = 0.6 \times 10^{-6} \times E_v \times R_1$$

Where, V_{iout} = IOUT output voltage [V]

E_v = lillumiance of the ALS(Ambient Light Sensor) surface [lx]

R_1 = IOUT output resistor [Ω]

2.6.2 인터페이스 설명



2.6.2.1. Power, Ready LED

System Ready (RDY): 시스템이 정상동작하고 있음을 알린다. (정상: 점멸 상태)

Power (PWR): 전압이 인가 되었음을 의미한다. (RED LED ON 상태)

2.6.2.2. Serial Port LED

핀이름	기 능	용 도
Debug Port	TxD	Debug Port 송신 상태 표시
	RxD	Debug Port 수신 상태 표시
COM Port 1 (RS232)	TxD	COM Port1 송신 상태 표시
	RxD	COM Port1 수신 상태 표시
COM Port 2 (RS232)	TxD	COM Port2 송신 상태 표시
	RxD	COM Port2 수신 상태 표시
COM Port 3 (RS422/RS485)	TxD	RS422의 경우, COM Port3 송신 상태 표시 RS485의 경우, 송·수신 상태를 표시한다.
	RxD	RS422의 경우, COM Port3 수신 상태 표시 RS485의 경우, 동작하지 않는다. (LED off)
COM Port 4 (RS422/RS485)	TxD	RS422의 경우, COM Port4 송신 상태 표시 RS485의 경우, 송·수신 상태를 표시한다.
	RxD	RS422의 경우, COM Port4 수신 상태 표시 RS485의 경우, 동작하지 않는다. (LED off)

2.6.2.3. GPIO LED

Eddy-S4M 이 제공하는 제공되는 GPIO 포트는 최대 34개 이다.

No	핀이름	용 도	I/O
1	PC25	GPIO 로만 사용가능	I/O
2	PC24	GPIO 로만 사용가능	I/O
3	PC20	GPIO or SPI1_NPCS3	I/O
4	PC19	GPIO or SPI1_NPCS2	I/O
5	PC18	GPIO or SPI1_NPCS1	I/O
6	PC17	GPIO 로만 사용가능	I/O
7	PC15	GPIO 로만 사용가능	I/O
8	PC14	GPIO 로만 사용가능	I/O
9	PC13	GPIO 로만 사용가능	I/O
10	PC12	GPIO 로만 사용가능	I/O
11	PC10	GPIO 로만 사용가능	I/O
12	PC9	GPIO 로만 사용가능	I/O
13	PC5	GPIO or SPI1_NPCS1	I/O
14	PC3	GPIO or AD3 or SPI1_NPCS3	I/O
15	PC2	GPIO or AD2 or PCK0	I/O
16	PC1	GPIO or AD1 or PCK0	I/O
17	PC0	GPIO or AD0	I/O
18	PB31	GPIO or PCK1	I/O
19	PB30	GPIO or PCK0	I/O

20	PB21	GPIO or RF0	I/O
21	PB20	GPIO or RK0	I/O
22	PB19	GPIO or RTD0 or TIOB5	I/O
23	PB18	GPIO or TD0 or TIOB4	I/O
24	PB17	GPIO or TF0 or TCLK4	I/O
25	PB16	GPIO or RxD5 or TCLK3	I/O
26	PB13	GPIO or RxD5	I/O
27	PB12	GPIO or TxD5	I/O
28	PB3	GPIO or SPI1_NPCS0 or TIOA5	I/O
29	PB2	GPIO or SPI1_SPCK	I/O
30	PB1	GPIO or SPI1_MOSI or TIOB3	I/O
31	PB0	GPIO or SPI1_MISO or TIOA3	I/O
32	PA30	GPIO 로만 사용가능	I/O
33	PA22	GPIO 로만 사용가능	I/O
34	PA5	GPIO 로만 사용가능	I/O

참고로 PIO 라인은 high-drive current capable 이 있어서 PC4-PC31(2mA)을 제외한 PIO 라인은 16mA를 드라이브 할 수 있다. (CPU Datasheet의 41.2 DC characteristics, 아래표 참조)

AT91SAM9260 DC Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Units
I _o	Output Current	PA0-PA31 PB0-PB31 PC0-PC3			16	
		PC4 - PC31 in 3.3V range			2*	mA
		PC4 - PC31 in 1.8V range			4	

* Eddy DK v2.1은 3.3V range 이므로 PC4-PC31 PIO는 2mA를 드라이브할 수 있다.

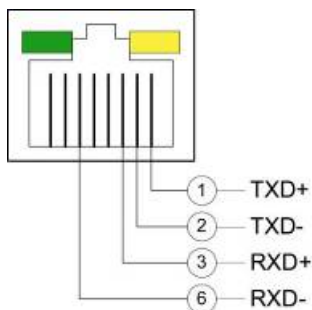
2.6.2.4. J10 : Ethernet

Eddy-S4M 모듈내 KSZ8041NL PHY가 내장되어 있기 때문에 트랜스 포머가 내장된 RJ45 커넥터만 연결하여 간단하게 Ethernet을 구현하였다.

WARNING : 트랜스포머가 내장된 RJ45 사용시 제품마다 내부회로가 다를 수 있다. 따라서 보드 설계시, 반드시 RJ45 커넥터 내부 회로의 핀번호를 확인하고 설계해야 한다.

KSZ8041NL의 기능을 아래와 같다.

- Fully compliant to IEEE 802.3u Standard
- Supports MDI/MDI-X auto crossover (Auto-MDI)
- MII interface support
- RMI interface support with external 50MHz system clock
- ESD rating (6kV)
- Built-in 1.8V regulator for core
- Available in 32-pin (5mm x 5mm) MLF® package



Pin	Signal	Description
1	TXD+	Physical transmit or receive signal (+ differential)
2	TXD-	Physical transmit or receive signal (- differential)
3	RXD+	Physical transmit or receive signal (+ differential)
6	RXD-	Physical transmit or receive signal (- differential)
LED	Description	

LAN Connection Speed

Left Green

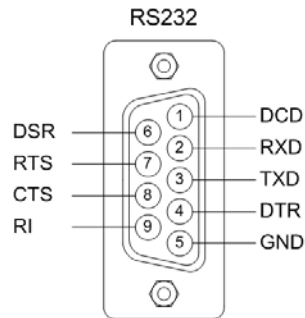
Speed	Pin State	LED Definition
10Base-T	H	OFF
100Base-TX	L	ON

LAN Connection Status

Right Yellow

Speed	Pin State	LED Definition
No Link	H	OFF
Link	L	ON
Activity	Toggle	Blinking

2.6.2.5. J17, 18 : COM Port 1 & Port 2



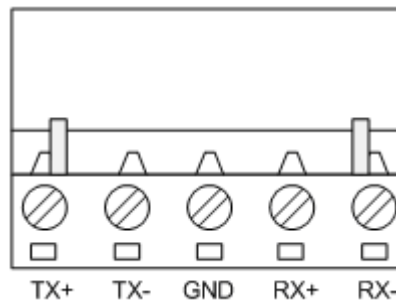
DB9 Male (COM Port 1, 2 공통)

RS232

Pin	Signal	Description
1	DCD	Data Carrier Detection (Input) (COM Port 1 only)
2	RXD	Receive Data (Input)
3	TXD	Transmit Data (Output)
4	DTR	Data Terminal Ready (Output) (COM Port 1 only)
5	GND	Ground
6	DSR	Data Set Ready (input) (COM Port 1 only)
7	RTS	Request to Send (Output)
8	CTS	Clear to Send (Input)
9	RI	Ring Indicator (Input)

* COM Port 2는 TxD, RxD, RTS, CTS 신호만을 제공한다.

2.6.2.6. J13, 14 : COM Port 3 & Port 4



RS422 Full Duplex

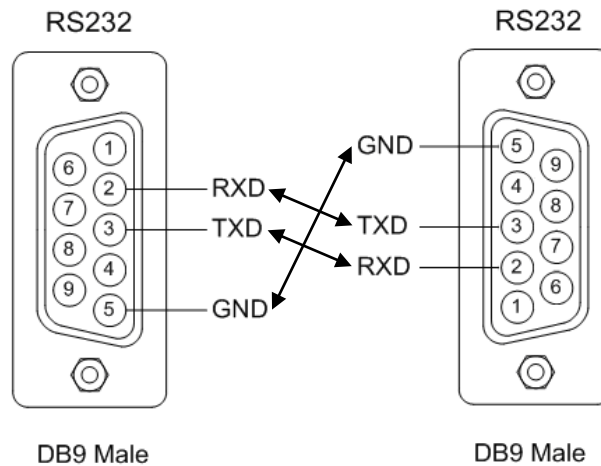
Pin	Signal	Description
1	TXD+	Transmit differential data positive (Output)
2	TXD-	Transmit differential data negative (Output)
3	GND	Ground
4	RXD+	Receive differential data positive (Input)
5	RXD-	Receive differential data negative (input)

RS485 Half Duplex

Pin	Signal	Description
1	TRX+	Transmit/Receive differential data positive
2	TRX-	Transmit/Receive differential data negative

2.6.2.7. J15 : Debug Port

디버그 포트를 통해서 제품의 디버그 메시지나 상태 정보를 확인 할 수 있다.



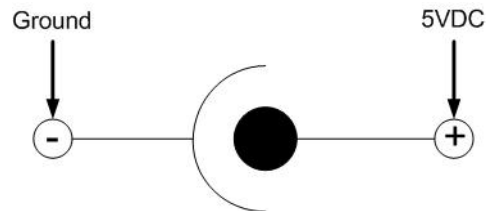
환경 설정

디버그 포트는 다음과 같이 환경이 설정이 되어 있으므로 사용자는 디버그 포트에 연결된 PC의 시리얼 포트에 설정을 다음과 같이 한다.

- Speed : 115200 bps
- Data bit : 8 bit
- Parity bit : Non Parity
- Stop bit : 1 bit
- Flow control : none

2.6.2.8. S1 : Power Jack

Contact	Polarity
Center (D : 2mm)	5VDC
Outer (D: 6.5mm)	Ground



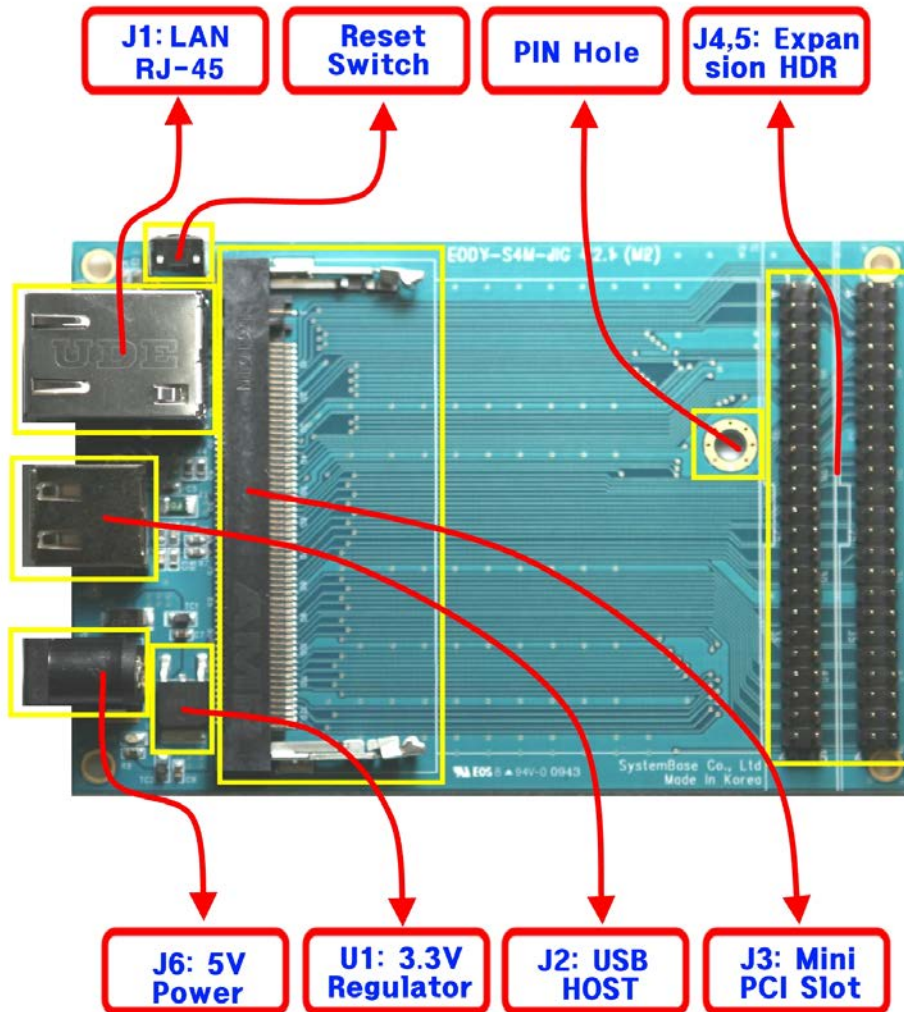
GPIO Connector pinout

Pin	Signal	Pin	Signal
1	PA5	2	PA22
3	PA30	4	NC
5	PB0	6	PB1
7	PB2	8	PB3
9	PB12	10	PB13
11	PB16	12	PB17
13	PB18	14	PB19
15	3.3V	16	3.3V
17	PB20	18	PB21
19	PB30	20	PB31
21	PC0	22	PC1
23	PC2	24	PC3
25	PC5	26	PC9
27	PC10	28	PC12
29	PC13	30	PC14
31	GND	32	GND
33	PC15	34	PC17
35	PC18	36	PC19
37	PC20	38	PC24
39	PC25	40	nRESET(IN)
41	RDY#	42	NRST(OUT)
43	TWCK	44	TWD

2.7 Eddy-S4M-JiG v2.1

Eddy-S4M 개발용 JIG 보드는 Eddy-S4M 을 탑재하여 프로그래머가 쉽게 자신의 어플리케이션을 탑재하고 테스트 할 수 있도록 도움을 주는 시험 보드이다.

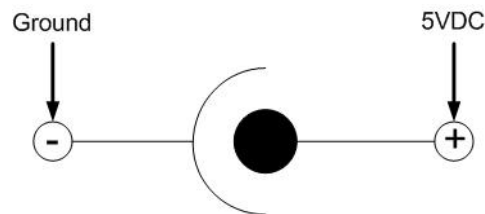
JIG 보드에는 Eddy-S4M 을 장착할 수 있는 miniPCI 커넥터, Ethernet RJ45, USB Host, Power, Reset Switch 가 포함되어 있으며, Eddy-S4M 에서 제공하는 모든 기능을 핀 커넥터 형태로 제공한다.



2.7.1

2.7.1 J6 : Power Jack

Contact	Polarity
Center (D : 2mm)	5VDC
Outer (D: 6.5mm)	Ground



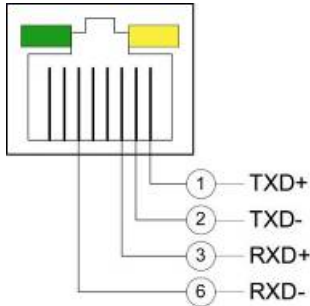
2.7.2 J1 : Ethernet

Eddy-S4M 모듈내 KSZ8041NL PHY가 내장되어 있기 때문에 트랜스 포머가 내장된 RJ45 커넥터만 연결하여 간단하게 Ethernet을 구현하였다.

WARNING : 트랜스포머가 내장된 RJ45 사용시 제품마다 내부회로가 다를 수 있다. 따라서 보드 설계시, 반드시 RJ45 커넥터 내부 회로의 핀번호를 확인하고 설계해야 한다.

KSZ8041NL의 기능을 아래와 같다.

- Fully compliant to IEEE 802.3u Standard
- Supports MDI/MDI-X auto crossover (Auto-MDI)
- MII interface support
- RMII interface support with external 50MHz system clock
- ESD rating (6kV)
- Built-in 1.8V regulator for core
- Available in 32-pin (5mm x 5mm) MLF® package



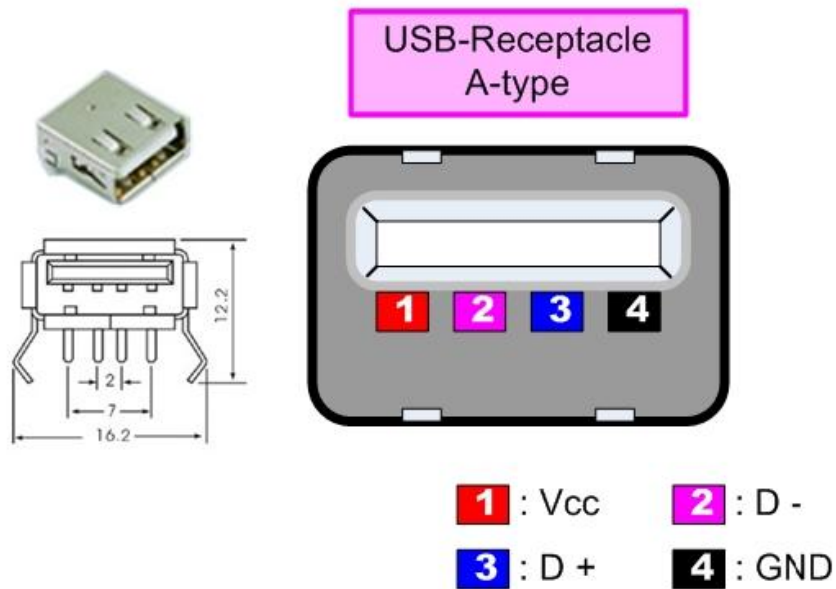
Pin	Signal	Description
1	TXD+	Physical transmit or receive signal (+ differential)
2	TXD-	Physical transmit or receive signal (- differential)
3	RXD+	Physical transmit or receive signal (+ differential)
6	RXD-	Physical transmit or receive signal (- differential)
LED	Description	

LAN Connection Speed			
Left Green	Speed	Pin State	LED Definition
	10Base-T	H	OFF
	100Base-TX	L	ON

LAN Connection Status			
Right Yellow	Speed	Pin State	LED Definition
	No Link	H	OFF
	Link	L	ON
	Activity	Toggle	Blinking

2.7.3 J2 : USB Host

Eddy-S4M 모듈에 있는 USB HUB Controller 출력에 연결되어 있고 핀 규격은 아래 그림과 같다.



2.7.4 RESET switch

핀이름	기 능	용 도	I/O
PC16	nRESET	Reset key 입력 신호를 지속적으로 polling 하여 "Low" 지속시간을 check 하고 아래와 같이 동작한다. 5 초미만 : 일반적인 reset 기능 5 초이상 : Factory Default 기능	IN

2.7.5 J4, 5 : Expansion Header

Eddy-S4M 에서 제공하는 대부분의 기능을 핀 커넥터 형태로 제공한다.
또한 Eddy-S4M-DK 보드에 직접 연결하여 기능을 확인할 수 있다.

J4

Pin	Signal	Pin	Signal
1	DTxD	2	DRxD
3	TxD0#	4	RxD0#
5	RTS0	6	CTS0
7	DTR0	8	DSR0
9	DCD0	10	RI0
11	TxD1#	12	RxD1#
13	RTS1	14	CTS1
15	3.3V	16	3.3V
17	P3_TX+	18	P3_TX-
19	P3_RX+	20	P3_RX-
21	P4_TX+	22	P4_TX-
23	P4_RX+	24	P4_RX-
25	DDM	26	DDP
27	DM2	28	DP2
29	DM3	30	DP3
31	GND	32	GND
33	DM4	34	DP4
35	SDDATA0	36	SDDATA1
37	SDDATA2	38	SDDATA3
39	SDCMD	40	SDCLK
41	SDCDN	42	SDWP
43	TWCK	44	TWD
45	RDY#	46	nRESET(IN)

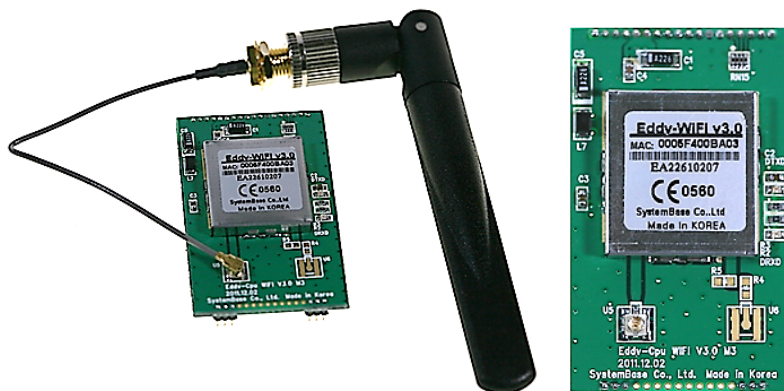
J5

Pin	Signal	Pin	Signal
1	LAN_TX+	2	LAN_RX+
3	LAN_TX -	4	LAN_RX-
5	LAN_LINK	6	LAN_Speed
7	PA5	8	PA22
9	PA30	10	NC
11	PB0	12	PB1
13	PB2	14	PB3
15	5V	16	5V
17	PB12	18	PB13
19	PB16	20	PB17
21	PB18	22	PB19
23	PB20	24	PB21
25	PB30	26	PB31
27	PC0	28	PC1
29	PC2	30	PC3
31	GND	32	GND
33	PC5	34	PC9
35	PC10	36	PC12
37	PC13	38	PC14
39	PC15	40	PC17
41	PC18	42	PC19
43	PC20	44	PC24
45	PC25	46	NRST(OUT)

2.8 Eddy-WiFi v3.0

(참고) Eddy-WiFi v2.1 관련 정보는 이전 매뉴얼을 참고하시기 바랍니다.

Eddy-WiFi 모듈을 Eddy-CPU v2.5 Series 에 연결하여 다양한 종류의 시리얼 장비를 (보안 장비, 통신관련 기기, 모뎀, 데이터 출력장치, 산업용 계측기 등) 무선 랜에 접속하여 사용할 수 있도록 할 수 있다. Eddy-WiFi 모듈은 IEEE 802.11b/g/n 무선 표준을 지원한다.



LEFT	Description
1	NA
2	NA
3	NA
4	NA

RIGHT	Description
1	NA
2	NA
3	NA
4	NA
5	VCC(3.3V)
6	VCC(3.3V)
7	USB Host Data(-)
8	USB Host Data(+)
9	NA
10	NA
11	H/W Reset
12	Ground
13	Ground
14	NA
15	NA
16	NA
17	NA
18	NA

2.9 Eddy-BT v2.1

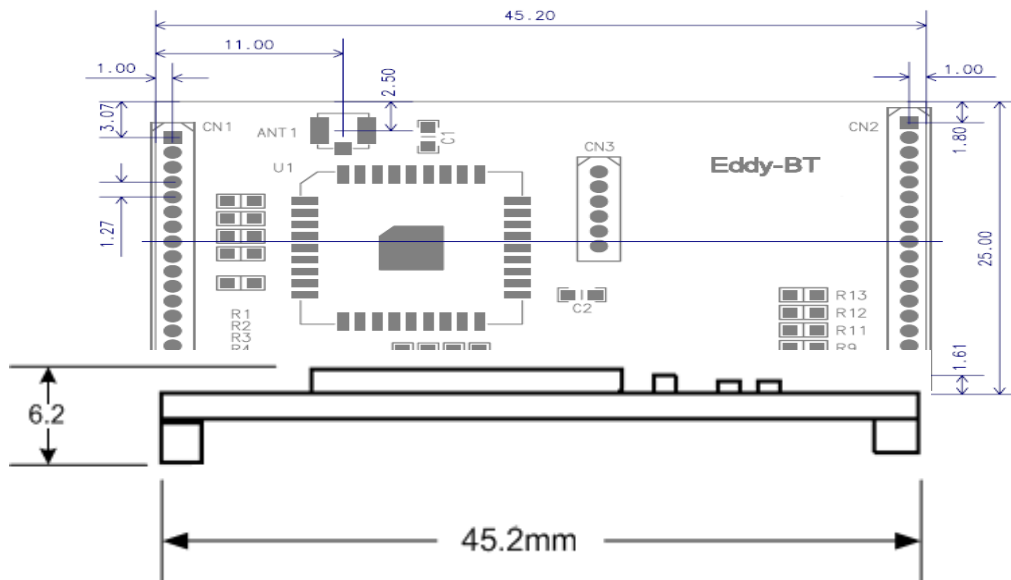
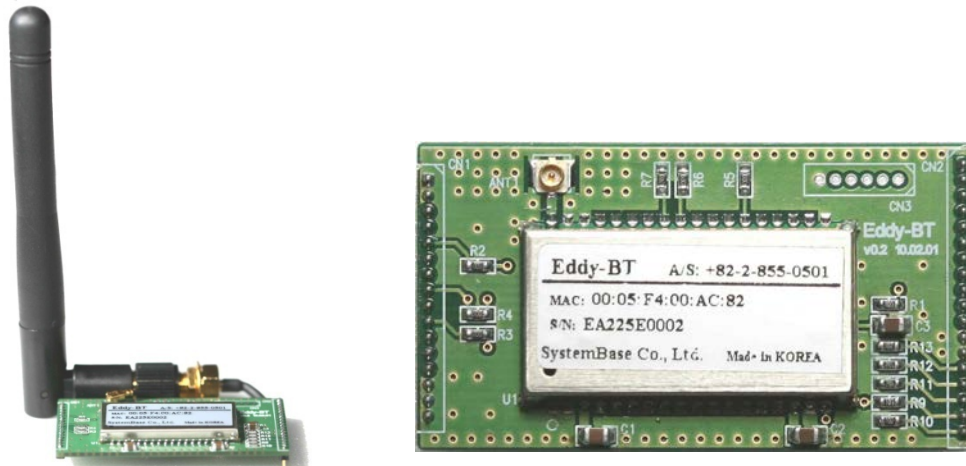
Eddy-BT 모듈은 Bluetooth 2.0 기반이며, 최대 1,000 m 의 통신거리를 지원하며, Eddy-CPU, Eddy-S4M 에 연결 되어 Bluetooth 통신방식으로 다양한 종류의 Bluetooth 디바이스 장비와 통신 가능한 모듈이다.

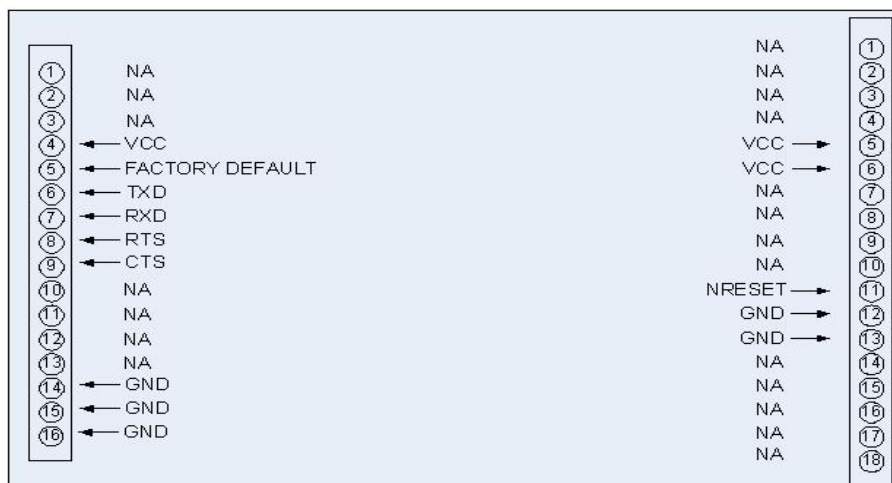
Eddy-BT 모듈의 통신 인터페이스는 시리얼 방식을 지원하며, Eddy-CPU, Eddy-S4M 에 연결시 시리얼 4 번째 포트 로 연결된다.

Eddy 의 기본 동작환경에는 Eddy-BT 를 사용하도록 고려되지 않아, HW Flow Control 을 사용할 경우 데이터 손실이 발생한다. (4 번째 포트는 Combo 방식으로 기본적으로 RS422 또는 RS485 를 지원하도록 구성되어 있어 RT S/CTS 신호선을 Auto Toggle 방식으로 사용하므로, 일반적인 RS232 방식의 HW 흐름제어로는 사용 할 수 없음)

Eddy-BT 를 올바르게 사용하려면, 제공되는 샘플소스코드 test_bluetooth.c 를 참조하기 바란다.

Eddy-BT 를 제어하기 위한 세부사항은 Eddy_User_Guide 의 7 장을 참조하기 바란다.

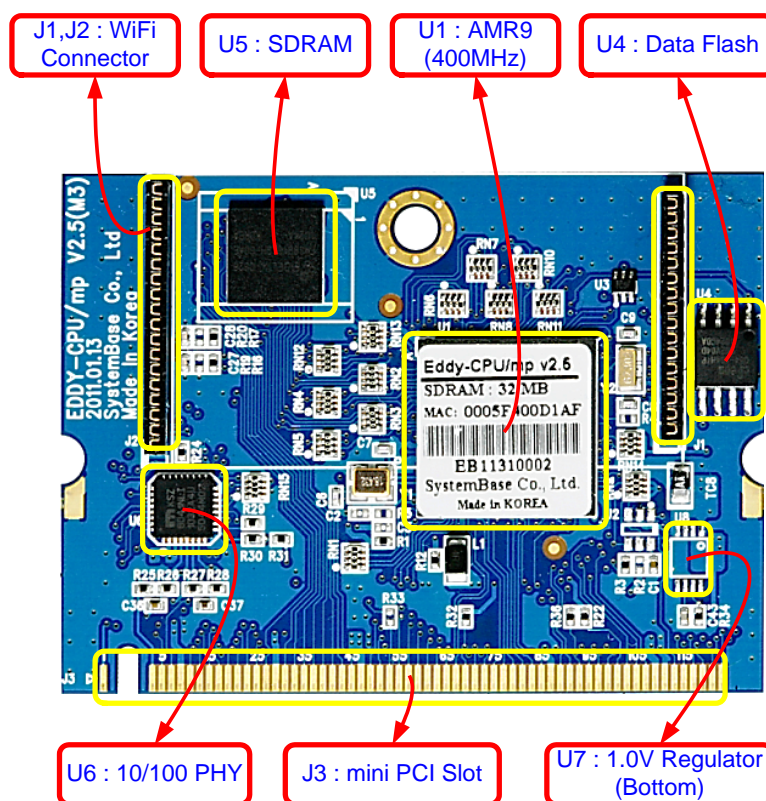




LEFT	Description
1	NA
2	NA
3	NA
4	VCC(3.3V)
5	Factory Reset
6	UART TXD
7	UART RXD
8	UART RTS
9	UART CTS
10	Pairing Signal
11	H/W Reset
12	NA
13	NA
14	Ground
15	Ground
16	Ground

RIGHT	Description
1	NA
2	NA
3	NA
4	NA
5	VCC(3.3V)
6	VCC(3.3V)
7	NA
8	NA
9	NA
10	NA
11	H/W Reset
12	Ground
13	Ground
14	NA
15	NA
16	NA
17	NA
18	NA

2.10 Eddy-CPU/mp v2.5



Eddy-CPU/mp V2.5 Mini PCI Card Type III System Connector Pinout(J3)

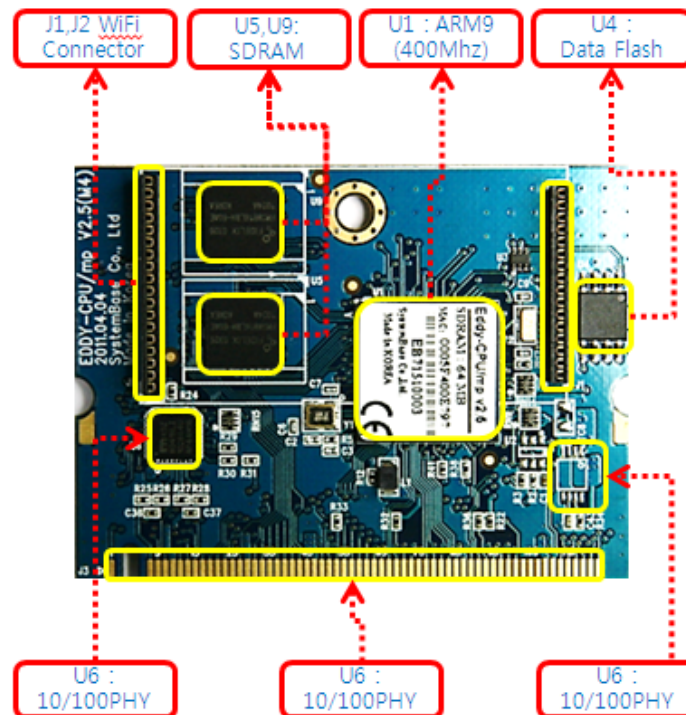
Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	LAN_RX+	2	LAN_TX+	63	PB8	64	PB9
	Key		Key	65	PB10	66	PB11
3	LAN_RX-	4	LAN_TX-	67	PB12	68	PB13
5	LAN_Speed	6	LAN_LINK	69	DRXD	70	DTXD
7	FPG	8	RDY#	71	PB16	72	PB17
9	3.3V	10	GND	73	PB18	74	PB19
11	D0	12	D1	75	PB20	76	PB21
13	D2	14	D3	77	PB22	78	PB23
15	D4	16	D5	79	PB24	80	PB25
17	D6	18	D7	81	PB26	82	PB27
19	D8	20	D9	83	PB28	84	PB29
21	D10	22	D11	85	PB30	86	PB31
23	D12	24	D13	87	3.3V	88	GND
25	D14	26	D15	89	PC0	90	PC1
27	NRD	28	NWE	91	PC2	92	PC3
29	3.3V	30	GND	93	PC5	94	PC8
31	A0	32	A1	95	PC9	96	PC10
33	A2	34	A3	97	PC12	98	PC13
35	A4	36	A5	99	PC14	100	PC15
37	A6	38	A7	101	nRESET	102	PC17
39	A8	40	A9	103	PC18	104	PC19
41	A10	42	A11	105	PC20	106	PC21

43	A12	44	A13	107	PC22	108	PC23
45	A14	46	A15	109	3.3V	110	GND
47	3.3V	48	GND	111	GND	112	PC26
49	PA4	50	PA22	113	TWCK	114	TWD
51	PA5	52	PA30	115	DDP	116	DDM
53	PA31	54	NRST	117	HDPB	118	HDPB
55	PB0	56	PB1	119	HDMA	120	HDMA
57	PB2	58	PB3	121	NAND_OE	122	A21
59	PB4	60	PB5	123	NAND_WE	124	A22
61	PB6	62	PB7				

J2	
Pin	Signal Name
1	PB0
2	PB1
3	PB2
4	PB3
5	3.3V
6	3.3V
7	BHDM, USB Host Data(-)
8	BHDP, USB Host Data(+)
9	PA31 / TXD4
10	PA30 / RXD4
11	NRST
12	GND
13	GND
14	PA9 / WPID0
15	PC6 / WPID1
16	PC7 / WPID2
17	NC
18	NC

J1	
Pin	Signal Name
1	NC
2	NC
3	3.3V
4	3.3V
5	PC25 / BT_Factory
6	PB10 / TXD3
7	PB11 / RXD3
8	PC8 / RTS3
9	PC10 / CTS3
10	PC24 / BT_MODE
11	NRST
12	GND
13	GND
14	NC
15	NC
16	NC

2.11 Eddy-CPU/mp v2.5 32bit



Eddy-CPU/mp v2,5 32bit Mini PCI Card Type III System Connector Pinout(J3)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	LAN_RX+	2	LAN_TX+	63	PB8	64	PB9
	Key		Key	65	PB10	66	PB11
3	LAN_RX-	4	LAN_TX-	67	PB12	68	PB13
5	LAN_Speed	6	LAN_LINK	69	DRXD	70	DTXD
7	FPG	8	RDY#	71	PB16	72	PB17
9	3.3V	10	GND	73	PB18	74	PB19
11	D0	12	D1	75	PB20	76	PB21
13	D2	14	D3	77	PB22	78	PB23
15	D4	16	D5	79	PB24	80	PB25
17	D6	18	D7	81	PB26	82	PB27
19	D8	20	D9	83	PB28	84	PB29
21	D10	22	D11	85	PB30	86	PB31
23	D12	24	D13	87	3.3V	88	GND
25	D14	26	D15	89	PC0	90	PC1
27	NRD	28	NWE	91	PC2	92	PC3
29	3.3V	30	GND	93	PC5	94	PC8
31	A0	32	A1	95	PC9	96	PC10
33	A2	34	A3	97	PC12	98	PC13
35	A4	36	A5	99	PC14	100	PC15
37	A6	38	A7	101	nRESET	102	PC17
39	A8	40	A9	103	PC18	104	PC19
41	A10	42	A11	105	PC20	106	PC21
43	A12	44	A13	107	PC22	108	PC23

45	A14	46	A15	109	3.3V	110	GND
47	3.3V	48	GND	111	GND	112	PC26
49	PA4	50	PA22	113	TWCK	114	TWD
51	PA5	52	PA30	115	DDP	116	DDM
53	PA31	54	NRST	117	HDP A	118	HDPB
55	PB0	56	PB1	119	HDMA	120	HDMB
57	PB2	58	PB3	121	NAND_OE	122	A21
59	PB4	60	PB5	123	NAND_WE	124	A22
61	PB6	62	PB7				

J2	
Pin	Signal Name
1	PB0
2	PB1
3	PB2
4	PB3
5	3.3V
6	3.3V
7	BHDM, USB Host Data(-)
8	BHDP, USB Host Data(+)
9	PA31 / TXD4
10	PA30 / RXD4
11	NRST
12	GND
13	GND
14	PA9 / WPID0
15	PC6 / WPID1
16	PC7 / WPID2
17	NC
18	NC

J1	
Pin	Signal Name
1	GND
2	GND
3	GND
4	3.3V
5	PC25 / BT_Factory
6	PB10 / TXD3
7	PB11 / RXD3
8	PC8 / RTS3
9	PC10 / CTS3
10	PC24 / BT_MODE
11	NRST
12	3.3V
13	3.3V
14	GND
15	GND
16	GND

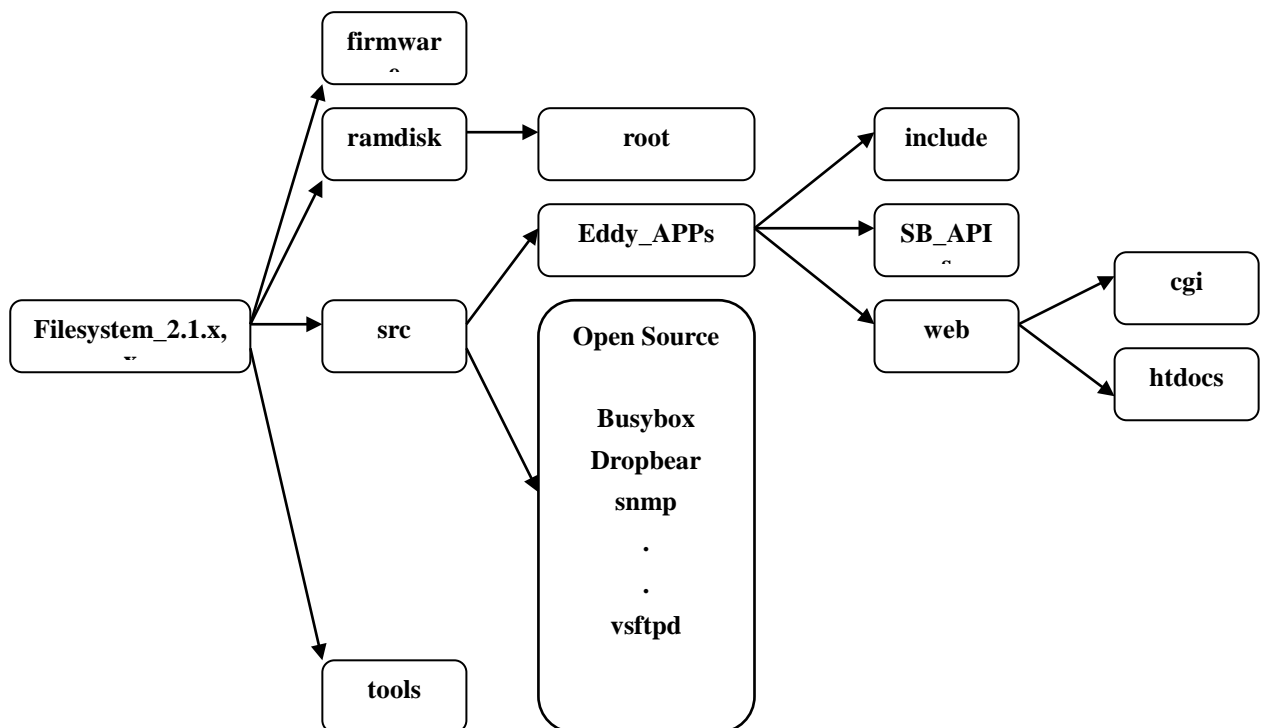
3장. 개발환경 구축

이 장에서는 사용자 응용프로그램을 작성하는 과정과 참고해야 할 사항에 대해 설명한다. SDK의 폴더 구성을 보면 아래 그림과 같다.

참고:

Eddy 에서 제공하는 소스 및 툴은 업데이트 시에 www.embeddedmodule.com 에 등록되므로 언제든지 새로운 버전을 다운로드 하여 설치 가능하다.

3.1 소스코드 폴더 구조



Firmware 폴더

부트로더, 커널, Filesystem 이미지, 등이 저장된 폴더

Ramdisk 폴더

Filesystem 이미지를 만들기 위한 폴더

root : Filesystem 에 들어가는 Eddy Filesystem 이 저장된 폴더

Tools 폴더

이미지 파일을 만들 때 사용되는 Tools 이 저장된 폴더

Src 폴더

Eddy 내에 포함된 어플리케이션들의 소스코드가 저장된 폴더

Src 에 대한 상세한 설명은 4장 응용 프로그램 컴파일에서 다루기로 한다.

Eddy_APPS 폴더

Eddy 에서 제공하는 기본 어플리케이션 소스코드가 있는 곳이며, 기타 폴더들은 Eddy 에서 실행 가능한 Open Source 코드의 소스이다.

3.2 사용 언어 (Language)

Eddy DK 의 응용프로그램은 C 언어로 작성되어야 한다. 본 제품 패키지와 함께 제공되는 예제 프로그램도 C 언어로 작성되어 있다. C 언어로 작업한다면 여러 개의 소스 파일을 통해 애플리케이션을 작성하여도 무방하다. 기존에 ANSI C 로 프로그래밍하는 데에 익숙하다면 Eddy 용 애플리케이션을 만드는 데에도 전혀 문제가 없을 것이다.

3.3 개발 환경

Eddy DK 로 개발을 하기 위해서는 Windows 호스트 또는 Linux 호스트 시스템이 필요하다.

다음은 테스트된 리눅스 버전 및 Windows 버전이다.

Windows	Linux
Windows XP SP2	Red Hat 9.0
Windows 2000	Fedora Core 4, 5, 6
Windows 2003	SUSE Linux Enterprise Server 10.2
	Ubuntu Linux 6.x, 7.x
	Debian Linuv 4.0
	CentOS 4.5
	Asianux edition 3

3.4 Windows 환경에 설치하기

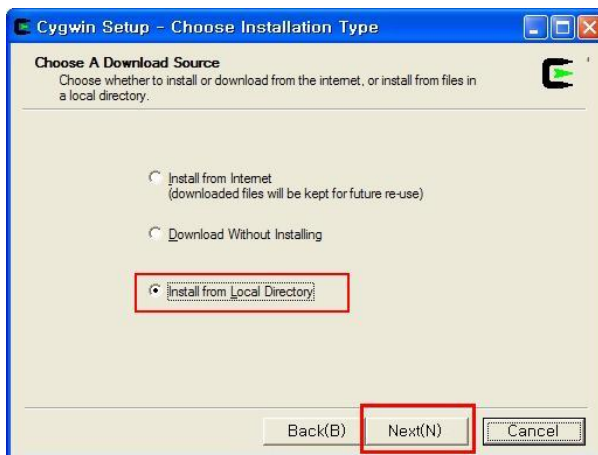
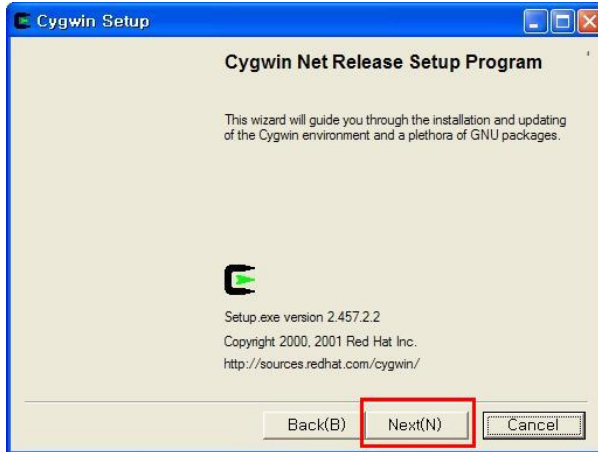
이 장에서는 Windows 호스트에 Eddy 개발환경을 설치하는 방법에 대해 설명한다. 이 매뉴얼에서는 Windows XP 를 기준으로 설명한다. 통합개발환경인 LemonIDE 를 사용하고자 하는 경우에는 "LemonIDE_User_Guide" 매뉴얼을 참조한다.

3.4.1 Cygwin 설치하기

Windows 환경에서 LemonIDE 를 실행하기 위해서는 리눅스 시스템에서 사용하는 일부 라이브러리가 필요하다.

Cygwin 은 Windows 에서 리눅스 환경이 호환되도록 하는 Windows 용 가상 리눅스 프로그램이다.

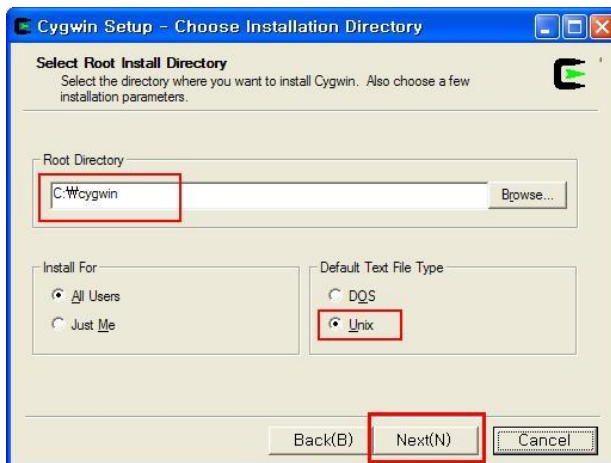
Cygwin 설치는 Eddy DK 에 제공되는 CD 의 SDK/Windows 폴더 밑에 Cygwin-Setup.zip 이며, 이 화일을 Windows PC 에 압축을 풀어 Setup.exe 를 실행하면, 다음과 같은 순서로 설치한다.



Cygwin 설치할 방법을 선택한다.

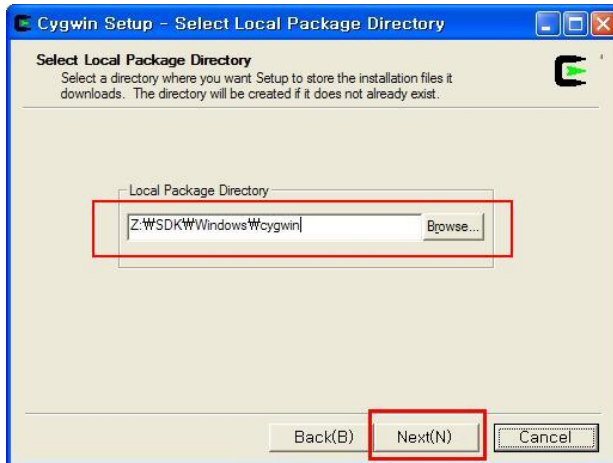
3 가지 설치 옵션 중 “Install From Local Directory”

를 선택하고, 다음을 클릭한다.



Cygwin 이 설치된 폴더를 선택한다.

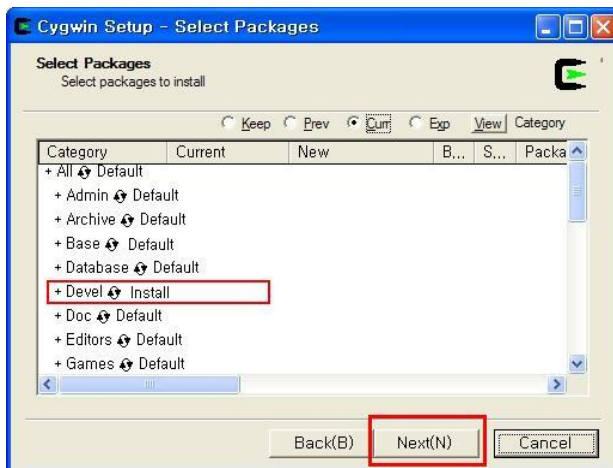
“C:\cygwin” 으로 설정한다.



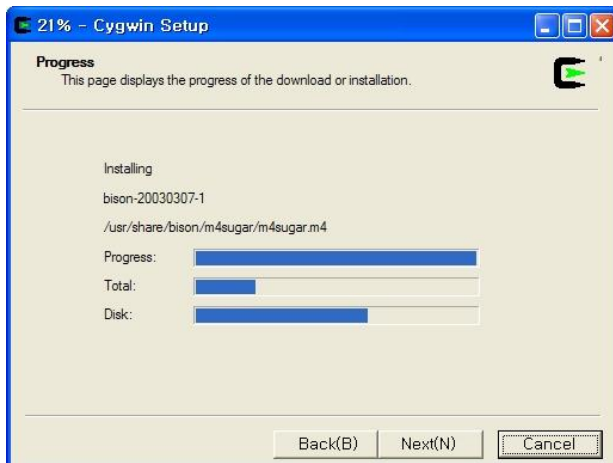
Cygwin-Setup.zip 앞축이 플러있는 폴더를 선택한다.

만약 C:\cygwin-Setup 폴더에 앞축을 풀어졌다면,

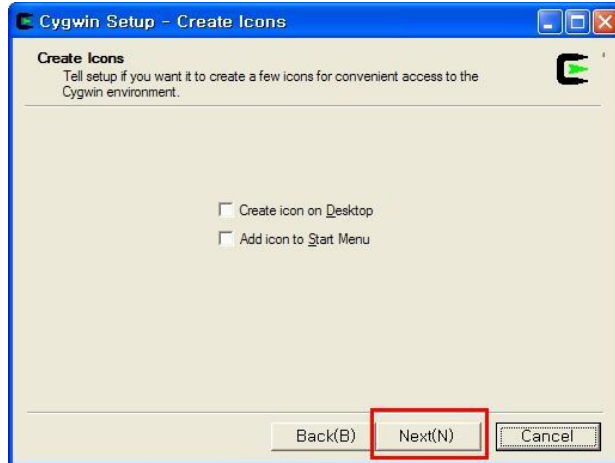
“c:\cygwin-setup” 폴더를 선택한다.



설치할 패키지를 선택한다. 좌측 화면에서 처럼 “Devel” 패키지 만 설치하도록 한다. 좌측 Default 를 클릭하면 Install 로 변경된다.



Cygwin 패키지 설치 중 화면



Cygwin 설치가 완료되면 설치를 종료한다.

3.4.2 Windows 환경변수 설정

Windows 환경에서 Eddy 에 필요한 라이브러리를 참조 가능하도록 Path 를 등록한다.

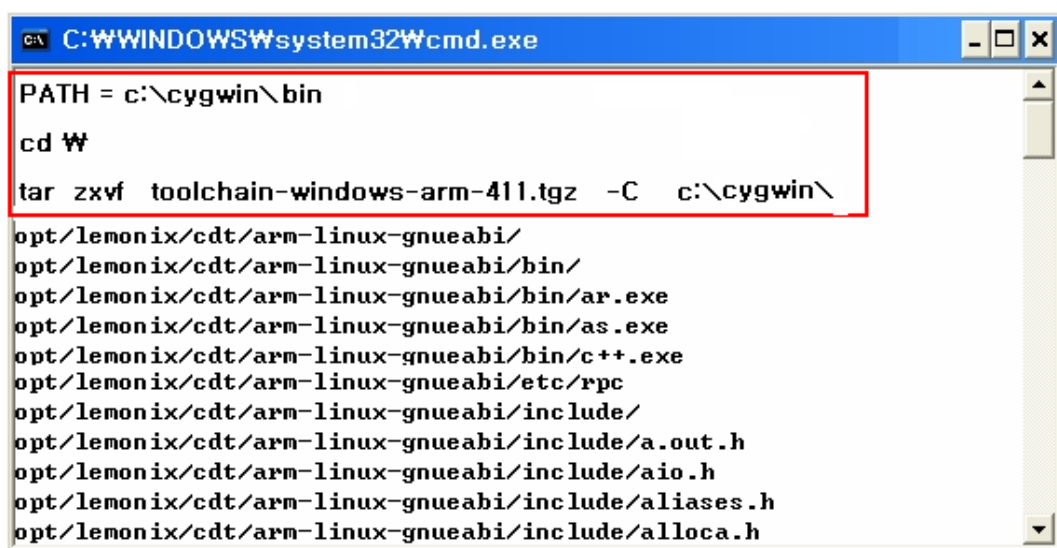
Windows 바탕화면 → 내컴퓨터 → 오른쪽 마우스로 클릭 → 속성 → 고급 탭 → 환경 변수를 선택하여 시스템 변수 중 Path 를 선택하여 편집을 클릭한 후 맨 앞에 다음을 추가한다.

`c:\cygwin\bin;`

3.4.3 ToolChain 설치하기

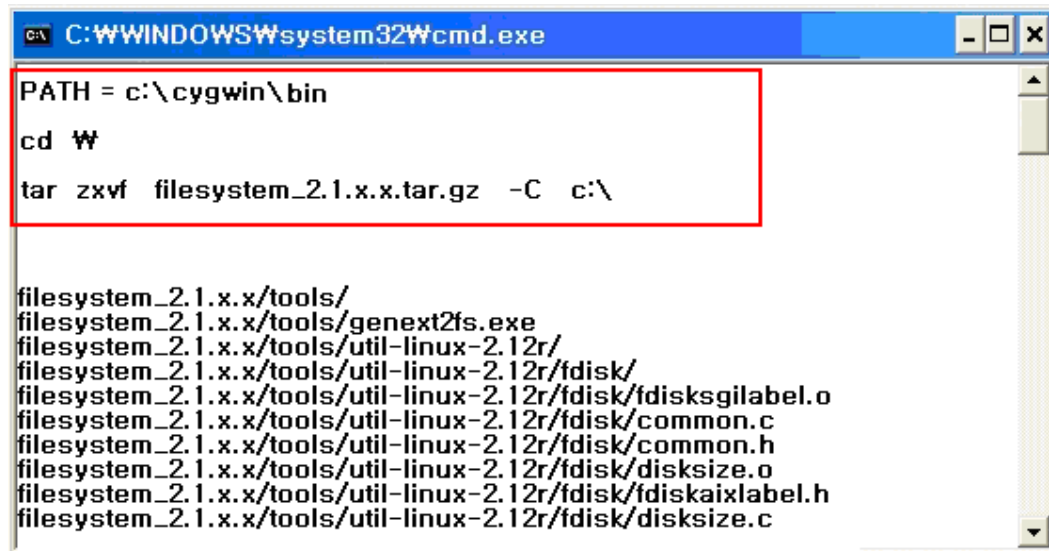
Windows 환경에서 작성한 코드를 Target 인 Eddy 에서 실행 가능하도록 컴파일 할 수 있는 Toolchain 을 설치한다. Toolchain 은 Eddy DK CD 의 SDK/Windows 폴더 밑에 “toolchain-windows-arm-411.tgz” 파일이다. 이 파일을 하드디스크 “C:” 드라이브의 루트 폴더에 복사한 다음 Windows 의 명령프롬프트 프로그램인 “CMD “ 를 실행하여 아래 그림과 같은 방식으로 압축을 푼다.

(대소문자 주의) ToolChain 이 설치되는 폴더는 “c:\cygwin\opt\lemonix\cdt” 이다.



3.4.4 Eddy DK Source 설치하기

Eddy DK 소스를 설치한다. DK Source 는 Eddy DK CD 의 SDK 폴더 밑에 “filesystem_2.x.x.x.tar.gz” 파일이다. 이 파일을 하드디스크 “C:” 드라이브의 루트 폴더에 복사한 다음 Windows 의 명령 프롬프트 프로그램인 “CMD” 를 실행하여 아래 그림과 같은 방식으로 압축을 푼다. (대소문자 주의) DK 소스가 설치되는 기본 폴더는 “c:\filesystem_2.x.x.x” 이다.



```
C:\WINDOWS\system32\cmd.exe

PATH = c:\cygwin\bin
cd W
tar zxvf filesystem_2.1.x.x.tar.gz -C c:\

filesystem_2.1.x.x/tools/
filesystem_2.1.x.x/tools/genext2fs.exe
filesystem_2.1.x.x/tools/util-linux-2.12r/
filesystem_2.1.x.x/tools/util-linux-2.12r/fdisk/
filesystem_2.1.x.x/tools/util-linux-2.12r/fdisk/fdisksgilabel.o
filesystem_2.1.x.x/tools/util-linux-2.12r/fdisk/common.c
filesystem_2.1.x.x/tools/util-linux-2.12r/fdisk/common.h
filesystem_2.1.x.x/tools/util-linux-2.12r/fdisk/disksize.o
filesystem_2.1.x.x/tools/util-linux-2.12r/fdisk/fdiskaixlabel.h
filesystem_2.1.x.x/tools/util-linux-2.12r/fdisk/disksize.c
```

3.5 Linux 환경에 설치하기

이 장에서는 Linux 호스트에 Eddy 개발환경을 설치하는 방법에 대해 설명하며, 이 매뉴얼에서는 Fedora Core 5 를 기준으로 한다. 통합 개발환경인 LemonIDE 를 사용하고자 하는 경우에는 “LemonIDE_User_Guide” 매뉴얼을 참조한다.

3.5.1 Toolchain 설치하기

Linux 환경에서 작성한 코드를 Target 인 Eddy 에서 실행 가능하도록 컴파일 할 수 있는 Toolchain 을 설치한다. ToolChain 설치 는 Eddy DK 에 제공되는 CD 의 SDK/Toolchain_Linux 폴더 밑에 “lemonide_linux_10x.tar.gz” 이며 ToolChain 이 설치되는 폴더는 /opt/lemonix 이다. (대소문자 주의)

참고:

설치하기의 모든 과정은 super user 권한으로 한다. 아래의 예제에서는 CDROM 을 /mnt/cdrom 에 마운트 했다고 가정한다.

만약 CDROM 을 다른 곳에 마운트 했다면 아래의 경로와 차이가 있을 수 있다.

```
# cd /
# tar -zxvf /mnt/cdrom/SDK/linux/lemonide*.tar.gz -C /
```

3.5.2 Eddy DK Source 설치하기

Eddy DK 전체 소스를 설치한다. Eddy DK Source 설치하는 Eddy DK 에 제공되는 CD 의 SDK/Source 폴더 밑에 “filesystem_2.x.x.x.tar.gz” 이다.

먼저 개발용 Linux PC 에서 작업 폴더로 이동하여 아래와 같이 설치하면 filesystem_2.x.x.x 폴더가 생성된다.

```
# pwd  
/home/shlee  
# tar -zxvf filesystem_2.x.x.x.tar.gz
```

압축을 풀면 filesystem_2.x.x.x 라는 폴더가 생성되면서 설치가 끝나며, 다음은 폴더의 내용이다.

3.6 개발환경 제거 방법

개발환경을 제거하기 위해서는 설치 파일이 있는 곳의 상위 폴더에서 설치 폴더를 삭제하면 된다.

3.6.1 Windows 환경에서의 환경 제거

Windows 환경에서는 DK Source 가 설치된 폴더와 Cygwin 이 설치된 폴더를 지우는 것으로 Eddy 개발환경을 간단히 삭제할 수 있다.

3.6.2 Linux 환경에서의 환경 제거

```
# rm -rf filesystem_2.x.x.x ; Eddy DK 소스 제거  
# rm -rf /opt/lemonix ; Eddy ToolChain 제거
```

4장. 응용 프로그램 컴파일

4.1 프로그램 종류

응용프로그램 작성하고 컴파일 하여 Eddy 에 탑재 실행해 보고, 이상이 없으면 최종 목적물인 펌웨어 이미지를 만들어 Eddy Flash 메모리에 저장하는 방법을 소개한다.

Eddy 에서 기본적으로 동작하는 응용 프로그램들은 Device Server 기능으로 구현되어 있다.

Device Server 로 동작하는 어플리케이션의 일부 소스는 공개하지 않는다.

제공되는 샘플 소스는 오픈소스 및 소켓과 시리얼 등 개발자가 참고하여 응용 가능하도록 최적화된 소스코드가 제공되므로 개발자는 이를 참고하여 Application 을 개발할 수 있다.

Open Source는 src 폴더 하위에 다음과 같이 존재한다.

폴더 이름	설명
busybox-1.5.0	shell 에서 사용되는 기본 명령어들을 포함하는 Linux Utility
dropbear-0.50	SSH (Secure Shell) 서버
gdbserver	LemonIDE 에서 사용되는 원격디버깅 프로그램 소스 코드를 공개 하지 않고 실행 파일만 제공된다.
mtt-util	Mtd 영역을 관리 프로그램
openssl-0.9.7c	SSL 의 일종인 OpenSSL 라이브러리
matrixssl-1-8-3	SSL 의 일종인 matrixssl 프로그램
thttpd-2.25b	HTTP 서버
vsftpd-2.0.5/	FTP 서버
ddns-1.8	DDNS 서버
ethtool-6	Ethernet 기반의 네트워크를 시험하는 프로그램
netkit-ftp-0.18	FTP 클라이언트
target-agent	LemonIDE 와 연동되어 유저 프로그램을 업로드, 실행 등의 기능을 하는 프로그램으로 소스코드가 공개되지 않는다.
net-snmp-5.4.1	SNMP V1/V2/V3 프로그램
Iptables-1.3.7	LAN 포트의 NAT 기능을 위한 브리지 프로그램
RT73	WiFi Device Driver
Wireless_tools.29	Wireless support Tool Applications
Wpa_supplicant-0.6.9	Wireless support Tool Applications
8712_linux_v2.6	RTL-8712 Device Driver

새 응용프로그램을 작성하는 경우에는 Eddy_APPS 폴더의 샘플 소스를 참조 하여 작성한다. Eddy_APPS 폴더 내의 프로그램 중 Device Server 어플리케이션 전용으로 사용되는 소스는 제공되지 않으므로, 샘플로 제공되는 다양한 용도의 소스코드를 참조하여 개발자가 원하는 응용 어플리케이션 개발이 가능하다.

파일 이름	설명	소스제공
eddy.c	Eddy 부팅 후 처음 실행되는 프로그램이며, 환경설정 정보대로 Eddy 가 실행 되도록 한다.	O
pinetd.c	Eddy 의 최상위 프로그램으로 하위 프로그램을 실행하고 감시한다. 새로운 Application 작성 시 이 곳에 등록하면 부팅 시 등록된 Application 이 실행된다.	O
com_redirect.c	Eddy 의 시리얼포트를 네트워크 상의 Windows PC 에서 자신의 Com 포트 인식하도록 지원하는 프로그램	X
tcp_client.c	서버에 접속하여 시리얼 포트와 소켓간의 상호 데이터를 교환하는 프로그램	X
tcp_server.c	소켓접속을 대기하여 시리얼 포트와 소켓간의 데이터를 교환하는 프로그램	X
detect.c	Portview 의 Detector 와 연동하는 프로그램 portview 매뉴얼 참조	X
portview.c	SystemBase 에서 제공하는 Windows 용 NMS 프로그램인 Portview 의 agent	X
tcp_broadcast.c	멀티 TCP 서버기능으로 최고 5 개의 client 접속을 지원하며 시리얼 데이터를 전체 Client 에 broadcasting	X
tcp_multiplex.c	멀티 TCP 서버기능으로 최고 5 개의 client 접속을 지원하며 시리얼 데이터를 개별 Client 에 전송하는 프로그램	X
udp.c	UDP 서버와 클라이언트 프로그램으로 UDP 소켓과 시리얼포트 간에 데이터를 교환하는 프로그램	X
wifi.c	WiFi Operating 소스 Flash 의 Config 화일을 읽어 등록된 설정대로 WiFi 를 동작하는 샘플소스	O
test_bluetooth.c	Bluetooth 응용 샘플 소스 Eddy-CPU, Eddy-S4M 에 연결된 Eddy-BT 모듈의 예제 샘플 소스	O
test_read_config.c	Flash Configuration Read/Write 샘플 소스	O
test_serial.c	시리얼 포트 응용 샘플 소스 Argument 로 지정한 포트번호를 오픈하여 연결된 상대방에서 수신한 데이터를 재 송신하는 샘플 소스	O
test_serial_to_lan-1.c	serial to lan 통신 응용 샘플 소스 Flash 의 Config 파일정보를 읽어 TCP 방식으로 소켓 접속을 대기하며, 접속 후 시리얼과 소켓포트간의 데이터를 상호 교환하는 샘플 소스	O
test_serial_to_lan-2.c	serial to lan 통신 응용 샘플 소스 Flash 의 Config 파일정보를 읽어 TCP 방식으로 등록된 서버에 접속을 시도하며, 접속 후 시리얼과 소켓포트간의 데이터를	O

	상호 교환하는 샘플 소스	
test_tcp_server.c	TCP 소켓 통신 응용 샘플 소스 argument 로 지정한 소켓번호로 TCP 접속을 대기하며, 접속 후 상대방 소켓에서 수신한 데이터를 상대방 소켓으로 다시 재 송신하는 샘플 소스	O
test_tcp_client.c	TCP 소켓 통신 응용 샘플 소스 argument 로 지정한 서버의 IP 주소와 소켓번호로 접속을 시도하며, 접속 후 상대방 소켓에서 수신한 데이터를 상대방 소켓으로 다시 재 송신하는 샘플소스	O
test_udp_server.c	UDP 소켓 통신 응용 샘플 소스 argument 로 지정한 소켓번호로 UDP 접속을 대기하며, 접속 후 상대방 소켓에서 수신한 데이터를 상대방 소켓으로 다시 송신하는 샘플소스	O
test_udp_client.c	UDP 소켓 통신 응용 샘플 소스 argument 로 지정한 서버의 IP 주소와 소켓번호로 UDP 접속을 시도하며, 접속 후 상대방 소켓에서 수신한 데이터를 상대방 소켓으로 다시 송신하는 샘플소스	O
def.c	Eddy 환경설정 프로그램 Telnet 으로 접속하여 Eddy 의 동작 환경을 설정할 수 있는 프로그램	O
upgrade.c	펌웨어 업데이트용 프로그램	O
testdk.c	Eddy-DK, Eddy-S4M-DK 시험 프로그램	O
ddns_agent.c	DDNS 서버에 Eddy IP 정보를 전달하는 프로그램	O
test_gpio_led.c	GPIO LED 테스트 프로그램 (Only Eddy-DK)	O
test_gpio_pin.c	GPIO Pin 테스트 프로그램 (Only Eddy-DK)	O
test_adc.c	ADC (Analog Disgital Converter) 테스트 프로그램	O
test_sio.c	시리얼 포트 테스트 프로그램	O
test_rtc.c	RTC (Real Time Clock) 테스트 프로그램	O
test_dio.c	DIO (Digital Input Output) 테스트 프로그램 (Only Eddy-DK)	O
test_keypad.c	Key Pad 테스트 프로그램 (Only Eddy-DK)	O
test_mmc.c	SD Memory 테스트 프로그램	O
test_lcd.c	LCD 테스트 프로그램 (Only Eddy-DK)	O
test_nand.c	NAND Flash 테스트 프로그램 (Only Eddy-DK)	O
test_spi_eeprom.c	SPI 인터페이스에 연결된 EEPROM 테스트 프로그램 (Only Eddy-DK)	O
/include	어플리케이션에서 필요한 헤더파일이 있는 폴더	O
/SB_APIs	Eddy 에서 제공하는 전용 라이브러리 폴더	X
/web	Eddy 에서 실행되는 Web 의 html 코드와 CGI 소스 폴더	O

4.2 응용 프로그램 작성

Eddy 에서 실행 가능한 응용 프로그램을 작성한다. 먼저 hello_world.c 파일을 src/Eddy_APPS 폴더에 다음과 같이 작성한다.

```
#include <stdio.h>

int main()
{
    While (1)
    {
        printf("hello world !!!\n");
        sleep (1);
    }
}
```

4.3 Makefile 작성 방법

응용프로그램을 컴파일 하려면, Eddy_APPS/Makefile 에 응용 프로그램 컴파일 관련 정보를 등록해야 한다. 다음은 src/Eddy_APPS/ 에 있는 Makefile 파일 내용이다. 아래의 그림은 응용 프로그램 컴파일 시 필요한 환경 값을 설정하는 부분이다. 아래의 붉은색으로 표시된 TARGET 에 이름을 추가하고, 컴파일 환경을 등록한다.

```
TARGET = eddy      pinetd      def      ddns_agent      \
upgrade  portview  upgradetftp  detect      \
tcp_server tcp_client tcp_multiplex tcp_broadcast      \
udp      rt_test   hello_world
udp : udp.o
    rm -f $@
    $(CC) $(CFLAGS) $(LDFLAGS) $(IFLAGS) -o $@ $$.o $(LIBS)
    $(STRIP) $@

Hello_World : Hello_World.o
Rm -f $@
$(CC) $(CFLAGS) $(LDFLAGS) $(IFLAGS) -o $@ $$.o
$(STRIP) $@
```

4.4 응용 프로그램 컴파일

응용 프로그램을 작성하고 Makefile 에 컴파일 환경을 등록했으면, Eddy 에서 실행가능 하도록 소스를 컴파일 한다.

4.4.1 Windows 환경에서의 컴파일

Windows 환경에서의 컴파일 작업은 cmd (command prompt)창을 통해 단순히 Makefile 이 존재하는 폴더에서 “make” 를 입력하는 것으로 완료된다. 아래 그림과 같이 컴파일이 정상적으로 완료되면 Hellow_World 라는 실행 파일이 생성된 것을 알 수 있다. 물론, 크로스 컴파일 작업으로 생성된 실행파일이므로, Windows 환경에서는 직접 실행은 해볼 수 없다. 이 프로그램을 Eddy 하드웨어에서 실행하려면, 실행 파일을 FTP 로 업로드하여 Eddy 에서 실행해야 한다. (FTP를 통해 업로드한 실행파일은 Eddy 에 영구적으로 저장되지는 않는다.)

이에 대해서는 다음 장 (Chapter 5. Firmware 만들기) 에서 설명한다.

```
C:\filesystem_2.1.x.x/src/Eddy_APPS> make hello_world
/opt/lemonix/cdt/bin/arm-linux-gcc -O2 -g -Wall -Wno-nonnull -c -o hello_world.o .....
/opt/lemonix/cdt/bin/arm-linux-gcc -L/opt/lemonix/cdt/lib -L/opt/lemonix/cdt/bin hello_world.o
-o hello_World
C:\filesystem_2.1.x.x/src/Eddy_APPS>
C:\filesystem_2.1.x.x/src/Eddy_APPS> ls
hello_world SB_APIs def.c eddy kt.c pinetd portview.o
tcp_client.c tcp_client tcp_multiplex.o . . .
```

4.4.2 Linux 환경에서의 컴파일

Linux 환경에서의 컴파일 작업은 Makefile 이 존재하는 폴더에서 “make” 를 입력하는 것으로 완료된다. 아래 그림과 같이 컴파일이 정상적으로 완료되면 Hellow_World 라는 실행 파일이 생성된 것을 알 수 있다. 물론, 크로스 컴파일 작업으로 생성된 실행파일이므로, Linux 환경에서는 직접 실행은 해볼 수 없다. 이 프로그램을 Eddy 하드웨어에서 실행하려면, 실행 파일을 FTP 로 업로드 하여 Eddy 에서 실행해야 한다. (FTP를 통해 업로드 한 실행파일은 Eddy 에 영구적으로 저장되지는 않는다.)

이에 대해서는 다음 장 (Chapter 5. Firmware 만들기) 에서 설명한다.

```
[shlee@localhost Eddy_APPS]$ make hello_world
/opt/lemonix/cdt/bin/arm-linux-gcc -O2 -g -Wall -Wno-nonnull -c -o hello_world.o hello_world.c
/opt/lemonix/cdt/bin/arm-linux-gcc -L/opt/lemonix/cdt/lib -L/opt/lemonix/cdt/bin hello_world.o
.....
[shlee@localhost Eddy_APPS]$ ls
Hello_World* SB_APIs/ def.c* eddy* kt.c pinetd* portview.o
server* tcp_client* tcp_multiplex.o tcps* upgrade* . . .
```

4.4.3 LemonIDE 를 통한 컴파일

LemonIDE 는 Eclipse 기반의 통합 개발환경으로 GUI 기반의 환경을 제공하여 보다 직관적인 개발이 가능하다. LemonIDE 는 Windows 환경과 Linux환경 모두에서 사용가능 하다. LemonIDE를 이용하면 소스의 코딩, 컴파일, 원격 디버깅은 물론 최종적인 펌웨어 이미지를 LemonIDE 내에서 개발이 가능 하도록 한다. LemonIDE 를 사용하려면 “LemonIDE_User_Guide” 메뉴얼을 참조하기 바란다.

4.5 Eddy 에서 실행하기

응용 프로그램을 실행하기 위해서는, 우선 이를 Firmware 로 만들어서 Eddy-CPU 의 플래시메모리 영역에 기록하여 전원을 리셋하여 부팅 후 실행되게 하는 방법이 일반적이나, 이는 약간의 시간이 소요되는 작업이기 때문에 개발 단계에서의 잦은 디버깅 과정에는 적합하지 않다.

디버깅 단계에서는 Eddy 에 내장된 FTP Server 를 통해 응용 프로그램의 실행 파일을 RAM방식의 파일 시스템에 업로드하고, Telnet 으로 접속하여 업로드된 응용 프로그램을 실행하여 문제가 없는지를 확인하는 방법이 있다. 다만, 이 방법은 개발 단계의 잦은 디버깅 작업에는 적합하지만, Eddy DK 의 전원을 리셋되면 응용 프로그램도 함께 사라진다는 단점이 있다.

LemonIDE 통합 개발환경에서는 이보다 더 진보된 방법을 제공한다. LemonIDE 에서 지원하는 디버깅 툴을 이용해 컴파일한 응용 프로그램을 Eddy 에 바로 전송하여 실행하고 결과를 바로 확인할 수 있다.

LemonIDE 를 사용하려면 “LemonIDE_User_Guide” 메뉴얼을 참조하기 바란다.

4.5.1 Eddy 에 업로드 하여 실행하기

응용 프로그램을 Eddy 에 업로드 하기 위해 FTP로 접속한다. Eddy의 ftp 서버에 접속하는 ID 와 password 는 telnet 접속 시 사용하는 것과 같다. 다음 예제는 실행파일 ‘hello_world’ 를 Eddy 의 /tmp 폴더로 업로드 하는 과정이다. 실행파일을 업로드 할 때에는 반드시 bin (바이너리 모드) 명령으로 업로드 해야 하며 업로드 하는 명령어는 “put <파일이름>” 명령을 사용한다. 아래의 그림은 Linux 환경에서 FTP 로 Eddy 에 접속하여 예제 프로그램을 업로드 하는 예제이다.

```
[shlee@localhost Eddy_APPS]$ ftp 192.168.0.223
Name (192.168.0.223:shlee): eddy
331 Please specify the password.
Password:
230 Login successful.
ftp> cd /tmp
ftp> bin
ftp> put hello_world
8914 bytes sent in 0.00027 seconds (3.3e+04 Kbytes/s)
ftp> bye
[shlee@localhost Eddy_APPS]$
```

Windows 환경에서는 cmd (Command Prompt) 창을 통해 Windows 에서 기본으로 제공하는 ftp 프로그램을 이용하여 업로드 한다. 전송이 완료되면 Eddy 에 접속된 Telnet 터미널에서 파일을 확인할 수 있다. 파일은 ‘chmod’ 명령으로 실행 가능한 모드로 바꿔야 실행이 가능해진다. 실행 가능 모드로 변경 후에 “./hello_world” 의 형태로 실행시켜 본다. 실행을 종료하려면 Ctrl+C 키로 프로그램 실행을 종료할 수 있다.

```
# ls
hello_world      login.id          tthttpd.log       login.pw
tthttpd.pid      utmp              . . .
#
# chmod 777 hello_world
#
# ./hello_world
Welcome to Eddy !
Welcome to Eddy !
Welcome to Eddy !
Welcome to Eddy !
```

4.5.2 Eddy 부팅 시 실행되도록 설정하기

응용 프로그램을 Eddy 에 업로드 하여 실행에 문제가 없다면, 펌웨어 이미지를 만들어 Eddy 의 Flash 메모리에 저장하여 Eddy 에 전원을 인가 시에 응용 프로그램이 작동하게 할 수 있다. 응용 프로그램이 Eddy 가 부팅 후 자동으로 실행되게 하려면 Eddy_APPS 폴더의 pinetd.c 에 응용 프로그램이 실행되도록 등록한다. 만일, 자동 실행이 필요치 않다면 이 단원을 작업할 필요는 없다.

```
//<=====
==
// Here User Application Launching !!
// -----
//
// ex) Task_Launch ("/sbin/hello", argument);
//           |           |
//           |           +---- Integer argument
//           +----- Application name with path
//
//=====
=>
Task_Launch ("/sbin/hello_world", 0);

signal(SIGCHLD, sig_chld);
```

Pinetd.c 소스를 수정하였으면 위 4.4 응용 프로그램 컴파일에서 처럼 “**make pinetd**” 를 실행하여 반드시 수정된 pinetd.c 소스를 재 컴파일 해야한다.

5장. Firmware 만들기

4장에서 응용 프로그램을 만들고 컴파일 하여 실행하는 과정을 예제 프로그램을 만들어서 설명하였다. 이 장에서는 예제 프로그램을 Eddy 에 영구적으로 저장할 수 있도록 Firmware 파일을 만들고 이를 Eddy 하드웨어에 적용하는 방법에 대해 소개한다.

5.1 Firmware 를 만드는 방법

filesystem_2.x.x.x/ramdisk 폴더에서 Eddy 의 Flash 메모리에 저장될 펌웨어 이미지 파일을 만들 수 있다. 펌웨어 이미지를 만들기 위해서는 filesystem_2.x.x.x/ramdisk 폴더에 있는 MakeFile 을 수정한다. Makefile 은 펌웨어 이미지를 만들 때 필요한 버전정보, 사용할 Ramdisk 의 크기, 복사할 Application 등의 정보를 설정할 수 있다.

참고:

DK Source 는 기본적으로 Linux 환경으로 배포된다. 그러므로 Windows 환경에서 Makefile 내의 일부 명령을 인식하지 못하는 경우가 있다. 이런 경우를 대비하여 Makefile 내에 아래의 그림에서 처럼 일부 유틸리티의 이름 뒤에 .exe 를 추가해 주어야 한다.

```
../tool/genext2fs → ../tool/genext2fs.exe  
../tool/mkimage → ../tool/mkimage.exe
```

```

IMAGE=ramdisk
FW_NAME = eddy-fs-2.x.x.x.bin → Firmware 이미지의 이름 및 버전정보

FIRMWARE_DIR = ../firmware → 작성된 펌웨어 이미지를 보관할 폴더

install:
#@echo "Making ramdisk image..."
#$(TOOL) -b 8192 -d root -D device_table.txt ramdisk
#../tool/genext2fs -U -b 5110 -d root -D device_table.txt ramdisk
#../tool/genext2fs -U -b 7158 -d root -D device_table.txt ramdisk
#../tool/mkcramfs -q -D device_table.txt root ramdisk
./tool/genext2fs.exe -U -b 10240 -N 1024 -d root -D device_table.txt ramdisk → Ramdisk 크기를
10,240 K 로 설정하며, Device_table.txt 를 참조하여 Eddy /dev 의 디바이스를 등록한다.
gzip -vf9 ramdisk
est -f ramdisk.gz
./tool/mkimage.exe -A arm -O linux -T ramdisk -C gzip -a 0 -e 0 -n $(FW_NAME) -d ./ramdisk.gz
$(FW_NAME)
test -f $(FW_NAME)
mv $(FW_NAME) $(FIRMWARE_DIR)/

release: → Eddy 에 복사될 Application 을 해당 폴더에 복사되도록 등록
cp -f ../src/Eddy_APPS/hello_world root/sbin
cp -f ../src/Eddy_APPS/eddy root/sbin
cp -f ../src/Eddy_APPS/com_redirect root/sbin
cp -f ../src/Eddy_APPS/tcp_server root/sbin
cp -f ../src/Eddy_APPS/tcp_client root/sbin
cp -f ../src/Eddy_APPS/tcp_broadcast root/sbin
cp -f ../src/busybox-1.5.0/busybox root/bin
cp -f ../src/dropbear-0.50/dropbear root/usr/local/sbin
cp -f ../src/dropbear-0.50/dropbearkey root/usr/local/sbin
cp -f ../src/ethtool-6/ethtool root/usr/local/sbin
cp -f ../src/net-snmp-5.4.1/agent/snmpd root/usr/local/sbin
.

```

Makefile 의 옵션에 의한 작업 내용은 다음과 같다.

Make release ; release 에 등록된 모듈을 ramdisk 영역으로 복사한다.

Make install ; Eddy 에서 사용할 Filesystem 을 펌웨어 이미지 파일로 생성한다.

Makefile 수정이 완료 되었으면 “ make release” 명령과 make install” 명령을 차례로 실행하여 Firmware 이미지 파일을 생성한다. 생성된 펌웨어는 Makefile 에 정의된 “FIRMWARE_DIR” 폴더에 저장된다. Windows 환경에서는 cmd (Command Prompt) 창을 Linux 방식과 같이 작업할 수 있다.

```
[shlee@localhost ramdisk]$ make release
.
.
.
[shlee@localhost ramdisk]$ make install
.
.
.
[shlee@localhost ramdisk]$ ls ../firmware
-rwxr-xr-x -----eddy-bl-2.x.x.x.bin
-rwxr-xr-x -----eddy-bs-2.x.x.x.bin
-rwxr-xr-x -----eddy-os-2.x.x.x.bin
-rwxr-xr-x -----eddy-fs-2.x.x.x.bin
.
.
```

위 그림에서 보는 바와 같이, Firmware 이미지 파일인 eddy-fs-2.x.x.x.bin 파일이 새로 생성 되었음을 알 수 있다. 이제, 이 Firmware 이미지를 Eddy에 WEB 또는 FTP 를 통해 업로드 한 후, Eddy 플래시 메모리에 저장하고 Eddy 를 리셋하게 되면 업로드한 펌웨어 환경으로 Eddy 가 실행되게 된다.

5.2 Firmware 업그레이드

생성된 펌웨어 파일을 Eddy 에 업로드 하여 Flash Memory 에 저장한다. 펌웨어 업그레이드 방법은 모두 4가지 방식을 지원한다.

FTP 방식	FTP 로 Eddy 에 접속하여 펌웨어 이미지를 업로드 한 다음 Telnet 으로 접속하여 Upgrade 명령으로 Flash 메모리에 저장하는 방식
Web 브라우저 방식	Eddy 의 WEB 서버에 접속하여 Upgrade 란을 통해 펌웨어 이미지를 Flash 메모리에 저장하는 방식으로 자세한 사용방법은 Eddy-User_Guide 를 참조한다.
부트로더 방식	Eddy DK 보드의 디버그 포트를 이용하여 재 부팅시 동작하는 부트로더를 통해 펌웨어 이미지를 Flash 메모리에 저장하는 방식이며 자세한 사용방법은 본 매뉴얼의 9 장 시스템 복구를 참조한다.
USB 방식	Eddy DK 보드의 USB Client 포트를 통해 펌웨어 이미지를 Flash 메모리에 저장하는 방식 (Windows Host 를 통해서만 가능) 으로 자세한 사용방법은 본 매뉴얼의 9 장 시스템 복구를 참조한다.

이 장에서는 FTP 를 이용한 업데이트 방법을 소개한다.

Windows 환경에서는 cmd (Command Prompt) 창을 통해 Windows 에서 기본으로 제공하는 FTP 프로그램을 이용하여 업로드 할 수 있다. 완성된 Firmware 이미지 파일인 eddy-fs-2.x.x.x.bin 를 ftp 를 이용하여 Eddy 의 /tmp 에 업로드 한다.

```
[shlee@localhost firmware]$ ftp 192.168.0.223
Connected to 192.168.0.223.
Name (192.168.0.223:shlee): eddy
331 Please specify the password.
Password:
230 Login successful.
ftp> cd /tmp
250 Directory successfully changed.
ftp> bin
200 Switching to Binary mode.
ftp> put eddy-fs-2.x.x.x.bin
local: eddy-fs-2.1.x.x.bin remote: eddy-fs-2.x.x.x.bin
227 Entering Passive Mode (192,168,0,223,195,50)
150 Ok to send data.
226 File receive OK.
2104287 bytes sent in 0.47 seconds (4.3e+03 Kbytes/s)
ftp> bye
221 Goodbye.
[shlee@localhost firmware]$
```

Eddy-에 텔넷으로 접속하여 /tmp 폴더에 'eddy-fs-2.x.x.x.bin' 파일이 확인한다.
'**upgrade eddy-fs-2.x.x.x.bin**' 란 명령으로 firmware 를 업데이트 한다.

```
# pwd
/tmp
# ls eddy-fs-2.x.x.x.bin
eddy-fs-2.x.x.x.bin
#
# upgrade eddy-fs-2.x.x.x.bin
FileSystem Erase ... 2388341 Bytes
FileSystem Write ... eddy-fs-2.x.x.x.bin, 2388341 Bytes
2388341 (2388341 bytes)
Flash Write OK
Flash Verify OK
...
```

업데이트 된 Firmware 가 동작하려면 시스템을 재 부팅 해야 한다. Eddy 가 재부팅 하면 Telnet 프로그램으로 Eddy 에 접속하여 응용 프로그램이 적재되어 있고, 자동으로 실행되었는지 확인해 볼 수 있다.

```
Eddy login: eddy
Password:
# ls /sbin
hello_world      ifconfig      nameif        switch_root
com_redirect     ifdown        pinetd        sysctl
.
# ps -ef
PID  USER  COMMAND
1    root  init
2    root  [posix_cpu_timer]
3    root  [softirq-high/0]
.
.
xx   root  /sbin/hello_world 1
```

응용 프로그램 실행의 출력결과는 시스템 표준 출력인 콘솔포트로만 출력된다. Eddy 의 콘솔포트는 Eddy DK 보드의 디버그포트이므로 telnet 에서는 응용 프로그램의 실행 결과를 볼 수 없다. Eddy DK 보드의 디버그포트를 PC 의 시리얼포트와 연결한 후 하이퍼터미널과 같은 시리얼 에뮬레이터 프로그램을 실행하고 통신속도를 115K, None, 8,1 로 설정하면 응용 프로그램의 실행 결과를 볼 수 있다.

```
Welcome to Eddy !
Welcome to Eddy !
Welcome to Eddy !
Welcome to Eddy !
Welcome to Eddy !
Welcome to Eddy !
Welcome to Eddy !
```

6장. 라이브러리 소개

본 절에서는, 사용자가 Eddy DK를 이용해서 프로그래밍 작업을 할 때 유용하게 이용할 수 있는 각종 API 들에 대해 소개한다.

6.1 사용하기 전에

이 장에서 소개하는 함수들은 모두 /src/Eddy_APPS/SB_APIs 폴더에 SB_APIs.a 로 포함된 API 들이며, Makefile 에 서도 마찬가지로 이에 대한 처리를 해 주어야 합니다. Eddy DK 와 함께 제공되는 예제 프로그램은 모두 이것을 이용 하게 되어 있으므로, 예제 프로그램의 소스 및 Makefile 을 참고 한다.

6.2 Makefile

Library 는 /src/Eddy_APPS/SB_APIs/ 폴더에 SB_API.a 라는 이름으로 존재한다. 이 라이브러리를 사용하려면 Makefile 에서 라이브러리 사용을 언급해야 하므로 /src/Eddy_APPS/ 폴더에 있는 Makefile 을 참조한다.

6.3 System 계열 함수

응용 프로그램을 작성하는데 필요한 Timer 와 Delay 기능에 대한 함수 들이다.

SB_GetTick

Function	Eddy 의 부팅 후 부터의 시간을 msec 단위로 반환한다.
Format	Unsigned long SB_GetTick (Void);
Parameter	None
Returns	0 ~ 4,294,967,295
Notice	리턴되는 값은 msec 단위의 시스템 Tick Counter 이며, Unsigned long 형의 최대값 0xffffffff 이후에는 다시 0 부터 시작한다. (약 50 일 간격으로 순환한다.)

SB_msleep

Function	지정한 msec 만큼 지연시킨다..
Format	void SB_msleep (int msec);
Parameter	msec 지연할 시간을 msec 단위로 설정한다.
Returns	none
Notice	정확히 msec 단위로 지연한다.

SB_AliveTime

Function	Eddy 가 기동한 시간부터 현재까지의 동작시간을 일,시간,분,초로 계산하여 반환한다.
Format	void SB_AliveTime (int *day, int *hour, int *min, int *sec);
Parameter	<div>*day 동작한 날짜 (0 ~)</div> <div>*hour 시간 (0 ~ 23)</div> <div>*min 분 (0 ~ 59)</div> <div>*sec 초 (0 ~ 59)</div>
Returns	None
Notice	

6.4 Eddy Environment 관련 함수

Eddy File System 에 관련된 환경관련 함수로서, Eddy 의 버전, 환경설정, 버전 등의 정보를 제공한다.

SB_GetVersion

Function	Eddy 에 포팅된 Bootloader, O/S, FileSystem 의 버전을 문자열 형태로 읽어온다.
Format	void SB_GetVersion (int type, char *version);
Parameter	<div>type 읽어올 버전을 지정한다.</div> <div> 'B' : Eddy 의 Bootloader 버전정보</div> <div> 'K' : Eddy 의 O/S 버전정보</div> <div> 'F' : Eddy 의 Filesystem 버전정보</div> <div>Version 버전정보 문자열을 저장할 데이터 포인터</div>
Returns	None
Notice	버전 정보는 문자열 형태로 "1.0a" 과 같이 읽어온다.

BootLoader 와 O/S 버전의 경우에는 시스템베이스가

제공하므로 변경될 수 없으며, FileSystem 버전은 사용자가 작성한 FileSystem 인 경우 직접 설정 가능하다.

Parameter type 가 'B' , 'K' , 'F' 가 아닌 것으로 호출하면 버전정보를 "0.00" 으로 리턴한다.

SB_ReadConfig

Function	Eddy 의 동작 환경설정 파일을 읽어온다.	
Format	void SB_ReadConfig (char *FileName, char *Dest, int Size);	
Parameter	FileName	읽어올 파일의 Path 를 포함한 파일이름
	*Dest	환경설정 파일을 읽어 저장할 버퍼의 포인트
	Size	읽어올 파일의 사이즈
Returns	Error Code	성공 시 1, 실패 시 -1 을 리턴
Notice	Eddy 에서 관리하는 환경설정 파일의 위치는/etc, /flash 에 위치한다. 사용자의 web 또는 telnet 에 의한 환경설정 변경 시 이곳에 저장되며, 모든 Eddy Application 은 이 환경파일 들을 참조하여 동작한다.	

SB_WriteConfig

Function	Eddy 의 동작 환경설정 정보를 파일로 저장한다.	
Format	void SB_WriteConfig (char *FileName, char *Source, int Size);	
Parameter	FileName	저장할 파일의 Path 를 포함한 파일이름
	Source	저장할 환경설정 정보가 저장된 struct 버퍼 포인트
	Size	저장할 struct 의 크기
Returns	Error Code	성공 시 1, 실패 시 -1 을 리턴
Notice		

SB_GetSharedMemory

Function	등록된 Shared Memory 의 포인터를 읽어온다.	
Format	void *SB_GetSharedMemory (int Key_ID, int Buffer_Size);	
Parameter	Key_ID	등록된 Shared Memory 의 ID
	Buffer_Size	사용한 Shared Memory 의 크기
Returns	*buffer_address	Shared Memory 의 메모리 번지

실패 시 -1 을 리턴한다.

Notice 이 함수는 Eddy 에서 실행하는 어플리케이션의 동작상태를 감시/제어하기 위해 제공하는 PortView 와 SNMP V1/V2/V3 agent 가 사용하는 공유 메모리로 각 어플리케이션에서는 이 번지에 자신의 상태정보를 주기적으로 갱신하여, 네트워크 상의 Portview 서버와 SNMP Server 에 정보를 제공한다.
단, 환경설정에서 PortView 와 SNMP Agent 가 설정되어 있어야 한다.

SB_SetSharedMemory

Function 사용할 Shared Memory 를 요청하고 메모리 포인터를 읽어온다.

Format void *SB_SetSharedMemory (int Key_ID, int Buffer_Size);

Parameter Key_ID 등록할 Shared Memory 의 ID
Buffer_Size 사용할 Shared Memory 의 크기

Returns *buffer_address Shared Memory 의 메모리 번지
실패 시 -1 을 리턴한다.

Notice Eddy 에서 이 함수는 PortView 와 SNMP agent 를 위해 사용한다.
사용자는 이를 응용 다른 목적을 위해 공유메모리를 사용 가능하다.

6.5 Serial 계열 함수

내장 시리얼 포트(UART)를 제어할 때 사용되는 함수들입니다.

SB_OpenSerial

Function 시리얼 포트를 오픈한다

Format int SB_OpenSerial (int Port_No);

Parameter Port_No 시리얼 포트 번호
0 : 첫번째 시리얼 포트
1 : 두번째 시리얼 포트
(Eddy-CPU, Eddy DK 인 경우에만 가능)

Returns -1 ~ N 오픈한 시리얼 포트의 핸들
-1 : 오픈에러
N : 오픈한 시리얼 포트의 핸들

Notice Eddy 는 최대 4 개의 시리얼포트를 제공하지만, Eddy-CPU 가 장착되는 일반 모델에는 1 개의 시리얼포트 만을 제공한다.
DK 보드는 4 개의 시리얼포트를 내장되어 있고, Eddy-CPU 난 Eddy DK 로 인식되도록 설정하면 4 개의 시리얼포트를 모두 사용

가능하다..

SB_InitSerial

Function	시리얼 포트의 데이터 통신 형식을 초기화한다.		
Format	Void SB_InitSerial (int Handle, char Speed, char LCR, char Flow);		
Parameter	Handle	OpenSerial 로 얻은 시리얼 포트의 핸들	
	Speed	통신 속도	
		0 : 150 BPS,	1 : 300 BPS
		2 : 600 BPS	3 : 1200 BPS:
		4 : 2400 BPS	5 : 4800 BPS
		6 : 9600 BPS	7 : 19200 BPS
		8 : 38400 BPS	9 : 57600 BPS
		10 : 115200 BPS	11 : 230400 BPS
		12 : 460800 BPS	13 : 921600 BPS
		LCR	X X P P S D D (8 bis binary)
		P P : Parity Bits	
		0 0 : None, 0 1 : Odd, 1 0, 1 1: Even	
		S : Stop Bits	
		0 : 1 bits, 1 : 2 bits	
		D D : Data Bits	
		0 0 : 5 bits, 0 1 : 6 bits	
		1 0 : 7 bits, 1 1 : 8 bits	
	FlowControl	흐름 제어 종류	
		0: no flow control	
		1: RTS/CTS flow control	
		2: Xon/Xoff flow control	
Returns	None		
Notice			

SB_SendSerial

Function	시리얼 포트를 통해 데이터를 출력한다.	
Format	Void SB_SendSerial (int handle, char *data, int length);	
Parameter	handle	시리얼 포트 또는 소켓을 오픈한 핸들
	data	출력할 데이터의 포인터
	length	출력할 데이터의 길이
Returns	None	
Notice	송신 버퍼가 Full 인 경우 20 msec 간격을 두고 최대 10 회 retry 하여, 출력을 완료 후 리턴한다.	

SB_ReadSerial

Function	시리얼 포트에서 데이터를 읽어온다.	
Format	int SB_ReadSerial (int handle, char *data, int length, int wait_msec);	
Parameter	handle	시리얼 포트를 오픈한 핸들
	data	읽어 들일 데이터를 저장할 버퍼 포인터
	length	버퍼의 메모리 크기(길이)
	wait_msec	수신 버퍼에서 데이터를 읽은 후 다음 수신 데이터를 대기할 시간
Returns	0 ~ n	읽어 들인 데이터의 길이
Notice	<p>wait_msec 를 0 으로 설정하면 시리얼 수신버퍼에 입력된 데이터만을 읽어오며, 0 이상의 값으로 설정하면, 수신버퍼에 입력된 데이터를 읽고, 지정한 msec 만큼 대기한 후 다음 시리얼 포트에 입력되는 데이터를 계속 읽어 하나의 패킷으로 읽어온다.</p> <p>읽어올 수 있는 최대 데이터 크기는 버퍼의 크기 즉 length 만큼이다. wait_msec 는 SB_GetDelaySerial 함수로 얻은 값으로 사용하거나 직접 계산하여 사용하도록 한다.</p>	

SB_GetMsr

Function 시리얼 포트에서 MSR register 값을 읽어온다.

Format Char SB_GetMsr (int handle);

Parameter handle 시리얼 포트를 오픈한 핸들

Returns Value MSR Register 값

Bit 7 6 5 4 3 2 1 0

Bit0: CTS change

Bit1: DSR change

Bit2: RI change

Bit3: DCD change

Bit4: CTS (0:Low, 1:High)

Bit5: DSR (0:Low, 1:High)

Bit6: RI (0:Low, 1:High)

Bit7: DCD (0:Low, 1:High)

Notice

SB_SetRts

Function 시리얼 포트의 RTS 신호선을 제어한다.

Format Void SB_SetRts (int handle, int value);

Parameter handle 시리얼 포트를 오픈한 핸들

Value 0: off RTS 신호를 Low 로 설정한다.

1: on RTS 신호를 High 로 설정한다.

Returns None

Notice

SB_SetDtr

Function	시리얼 포트의 DTR 신호선을 제어한다.
Format	Void SB_SetDtr (int handle, int value);
Parameter	handle 시리얼 포트를 오픈한 핸들 Value 0: off DTR 신호를 Low 로 설정한다. 1: on DTR 신호를 High 로 설정한다.
Returns	None
Notice	

6.6 Ethernet 계열 함수

현재 Eddy이 사용하는 네트워크와 관련된 정보를 입출력 합니다. 이 함수는 Eddy 의 최적화된 소켓 API 이며, 사용자는 이 함수 대신 POSIX 호환 표준 소켓 API 를 사용하여 개발할 수 있다.

SB_GetIp

Function	Eddy 에 할당된 IP 주소를 읽어온다.
Format	Unsigned int SB_GetIp (char *interface);
Parameter	Interface 네트워크 인터페이스 이름 WAN 포트는 "eth0" , LAN 포트는 "eth1" 설정한다.
Returns	Unsigned int IP 주소를 Unsigned int 형으로 반납한다.
Notice	Eddy 에 설정된 IP 주소를 읽어오는 것이 아니라, 동작중인 IP 주소를 읽어온다. Eddy 가 DHCP Client 로 동작 중인 경우 DHCP Server로부터 할당받은 네트워크 IP 주소를 읽어온다. IP 주소를 스트링 문자열로 변환하려면 아래를 참조한다. struct in_addr addr; addr.s_addr = SB_GetIp (); printf ("IP Address : %s ", inet_ntoa(addr));

SB_GetMask

Function	Eddy 에 할당된 Subnet Mask 주소를 읽어온다.
Format	Unsigned int SB_GetMack (char *interface);

Parameter	Interface	읽어 오고자 하는 인터페이스 이름 WAN 포트는 “eth0” , LAN 포트는 “eth1” 설정한다.
Returns	Unsigned int	Mask 주소를 unsigned int 형으로 반납한다.
Notice	SB_GetIp 와 참조	

SB_GetGateway

Function	Eddy 에 할당된 Gateway 주소를 읽어온다.
Format	Unsigned int SB_SetGateway(void);
Parameter	None
Returns	Unsigned int Gateway 주소를 unsigned int 로 반납한다.
Notice	SB_GetIp 와 참조

SB_ConnectTcp

Function	TCP 소켓으로 지정한 서버에 접속한다.
Format	Int SB_ConnectTcp (char *IP_Address, int Socket_No, int Wait_Sec, Int Tx_Size, int Rx_Size);
Parameter	IP_Address 접속할 서버의 IP 주소 문자열 Socket_No 접속할 서버의 소켓번호 Wait_Sec 접속 대기시간 (초 단위) Tx_Size 소켓의 Tx 버퍼 사이즈 (K bytes 단위) Rx_Size 소켓의 Rx 버퍼 사이즈 (K bytes 단위)
Returns	-1 ~ N 연결된 소켓의 핸들번호 -1 : 연결 실패 N : 연결된 핸들번호
Notice	접속요청 즉시 연결이 안되면 Wait_sec 로 지정한 시간만큼 재접속을 대기한 후 리턴한다. Tx,Rx_Size 는 소켓의 버퍼 사이즈로 최소 1 ~ 64 까지 설정 가능하다. 1 보다 작은 값을 입력하면 디폴트 4 kbytes 로 처리되며, 64 보다 큰 수를 입력하면 64 kbytes 로 처리된다.

SB_ListenTcp

Function	TCP 소켓으로 접속을 대기한다.	
Format	Int SB_ListenTcp (int Socket_No, Int Tx_Size, int Rx_Size);	
Parameter	Socket_No	접속을 대기할 TCP 소켓번호
	Tx_Bytes	소켓의 Tx 버퍼 사이즈 (K bytes 단위)
	Rx_Bytes	소켓의 Rx 버퍼 사이즈 (K bytes 단위)
Returns	-1 ~ N	TCP 접속을 대기할 소켓의 핸들번호
	-1	소켓 접속 대기 실패
	N	접속을 대기하는 TCP 소켓 핸들번호
Notice	<p>이 함수는 Non Blocking 함수로 접속을 요청한 후 접속대기를 하지 않고 즉시 리턴한다. 접속 대기는 SB_AcceptTcp 에서 처리된다.</p> <p>Tx,Rx_Size 는 소켓의 버퍼 사이즈로 최소 1 ~ 64 까지 설정 가능하다.</p> <p>1 보다 작은 값을 입력하면 디폴트 4 kbytes 로 처리되며, 64 보다 큰 수를 입력하면 64 kbytes 로 처리된다.</p>	

SB_AcceptTcp

Function	TCP 소켓 핸들의 네트워크 접속을 대기한다.	
Format	Int SB_AcceptTcp (int Socket_No, int wait_msec);	
Parameter	Socket_No	접속을 대기하는 TCP 소켓 핸들번호 (SB_ListenTcp 의 리턴 값)
	wait_msec	접속 대기시간 (단위 msec)
Returns	-1 ~ N	TCP 소켓으로 접속된 새로운 핸들번호.
	-1	소켓 에러
	0	접속 대기 중
	N	접속된 TCP 소켓의 새로운 핸들번호
Notice	<p>접속이 완료되어 새로운 핸들번호가 부여되면, 대기하는 이전 핸들은 이 함수 내에서 Close 된다.</p>	

SB_AcceptTcpMulti

Function	접속 대기하는 TCP 소켓 핸들의 네트워크 다중접속을 허가한다.	
Format	Int SB_AcceptTcpMulti (int Socket_No, int wait_msec);	
Parameter	Socket_No	접속을 대기하는 TCP 소켓 핸들번호 (SB_ListenTcp 의 리턴 값)
	wait_msec	접속 대기시간 (단위 msec)
Returns	-1 ~ N	TCP 소켓으로 접속된 새로운 핸들번호.
	-1	: 소켓 에러
	0	: 접속 대기 중
	N	: 접속된 TCP 소켓의 새로운 핸들번호
Notice	접속이 완료되어 새로운 핸들번호가 부여되면, 대기하는 이전 핸들을 닫지 않으므로 최대 1024 개의 소켓접속을 허가 할 수 있다.	

SB_ReadTcp

Function	접속된 TCP 소켓으로부터 데이터를 읽어온다..	
Format	Int SB_ReadTcp (int Handle, char *Buffer, int Buffer_Size);	
Parameter	Handle	TCP 소켓으로 접속된 핸들번호
	Buffer	읽어올 패킷 데이터를 저장할 버퍼 포인트
	Buffer_Size	저장할 버퍼의 크기
Returns	-1 ~ N	읽어온 데이터의 길이.
	-1	: 소켓 에러
	0	: 읽어온 데이터 없음
	N	: 읽어온 데이터의 길이
Notice	리턴 코드가 -1 이면, 접속된 상대와의 접속이 끊어진 상태이므로, TCP 소켓 핸들을 close 해야 한다.	

SB_CloseTcp

Function	TCP 소켓 핸들을 Close 한다.	
Format	Int SB_CloseTcp (int Handle);	
Parameter	Handle	Close 할 TCP 소켓 핸들번호

Returns	None
Notice	소켓핸들을 Shutdown 하여 통신을 종료하고 Close 한다.

SB_BindUdp

Function	UDP 소켓을 바인딩한다.	
Format	Int SB_BindUdp (int Socket_No);	
Parameter	Socket_No	바인드할 UDP 소켓번호
Returns	Handle	바인드된 UDP 소켓의 핸들번호 -1: 바인드 실패 N : 바인드된 UDP 소켓의 핸들번호
Notice		

SB_ReadUdp

Function	네트워크에서 바인드 된 UDP 소켓으로 송신한 데이터를 읽어온다.	
Format	Int SB_ReadUdp (int Handle, char *Buffer, int Buffer_Size);	
Parameter	Handle	UDP 소켓으로 바인드된 핸들번호
	Buffer	읽어올 패킷 데이터를 저장할 버퍼 포인트
	Buffer_Size	저장할 버퍼의 크기
Returns	-1 ~ N	읽어온 데이터의 길이. -1 : 소켓 에러 0 : 읽어온 데이터 없음 N : 읽어온 데이터의 길이
Notice	이 함수는 네트워크상의 상대방부터 바인드된 UDP 소켓으로 데이터를 보내오는 경우 내부에서 상대의 IP 주소와 소켓번호를 기억하여 SB_SendUdpServer 에서 사용하도록 한다.	

SB_SendUdpServer

Function	UDP 소켓으로 데이터를 송신한다. (Server 모드)	
Format	Int SB_SendUdpServer (int Handle, char *Buffer, int Data_Size);	
Parameter	Handle	UDP 소켓으로 바인드된 핸들번호
	Buffer	송신할 데이터가 저장된 버퍼 포인트
	Data_Size	송신할 데이터 크기

Returns	None
Notice	<p>이 함수는 네트워크로부터 Eddy 에 바인드된 UDP 소켓으로 먼저 데이터를 전송하여 상대의 네트워크 정보가 확인 된 후 사용이 가능하다. 즉 SB_ReadUdp 가 선행되어야 한다.</p> <p>먼저 데이터를 전송을 해야 하는 경우라면 SB_SendUdpClient 함수를 사용해야 한다.</p>

SB_SendUdpClient

Function	UDP 소켓으로 데이터를 송신한다. (Client 모드)										
Format	Int SB_SendUdpClient (int Handle, char *Buffer, int Data_Size, Char *IP_Address, int Socket_No);										
Parameter	<table> <tr> <td>Handle</td><td>UDP 소켓으로 바인드된 핸들번호</td></tr> <tr> <td>Buffer</td><td>송신할 데이터가 저장된 버퍼 포인트</td></tr> <tr> <td>Data_Size</td><td>송신할 데이터 크기</td></tr> <tr> <td>IP_Address</td><td>데이터를 전송할 상대의 IP 주소</td></tr> <tr> <td>Socket_No</td><td>데이터를 전송할 상대의 소켓번호</td></tr> </table>	Handle	UDP 소켓으로 바인드된 핸들번호	Buffer	송신할 데이터가 저장된 버퍼 포인트	Data_Size	송신할 데이터 크기	IP_Address	데이터를 전송할 상대의 IP 주소	Socket_No	데이터를 전송할 상대의 소켓번호
Handle	UDP 소켓으로 바인드된 핸들번호										
Buffer	송신할 데이터가 저장된 버퍼 포인트										
Data_Size	송신할 데이터 크기										
IP_Address	데이터를 전송할 상대의 IP 주소										
Socket_No	데이터를 전송할 상대의 소켓번호										
Returns	None										
Notice	<p>이 함수는 UDP 소켓으로 전송할 목적지 네트워크 정보를 알고 있는 경우에 사용 가능하다.</p> <p>먼저 데이터를 전송을 해야 하는 경우라면 SB_SendUdpClient 함수를 사용해야 한다.</p>										

6.7 GPIO Ioctl 함수

Eddy-CPU (최대 56 개), Eddy-S4M (최대 34 개) 의 GPIO 포트를 제어하는 함수이다.

GPIO 개별 포트를 통해 3.3V 의 전압을 감지하거나 출력을 제어할 수 있다. Eddy 에서 제공하는 Pin 들은 각종 디바이스 장치의 제어용으로 사용될 수 있는 공용이므로, GPIO 전용으로 사용되지는 않는다. Eddy 에서는 Port A, B, C 3 개의 포트그룹으로 각 32 개의 신호선을 제공한다.

Port A,B,C 각 포트에는 Eddy 에서 디바이스로 사용할 수도 있고 GPIO 로도 사용할 수 있도록 선택이 가능하며. 기본적으로 Eddy 는 Web 을 통해 환경을 설정할 수 있다. 자세한 사용방법은 Eddy_APPS 폴더에 있는 “testdk.c” 샘플 소스를 참조한다.

Eddy-CPU 의 GPIO 구성표

bytes	3								2								1								0							
bits	3 1	3 0	2 9	2 8	2 7	2 6	2 5	2 4	2 3	2 2	2 1	2 0	1 9	1 8	1 7	1 6	1 5	1 4	1 3	1 2	1 1	0 0	0 9	0 8	0 7	0 6	0 5	0 4	0 3	0 2	0 1	0 0
bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Port A											*																					
Port B	KEY	KEY	S1	S1	S0	S0	S0	S0	S0	S0	KEY	KEY	*	*	*	*	DEBB	DEBB	*	*	S3	S3	S2	S2	S1	S1	S0	S0	EEPROM	EEPROM	EEPROM	EEPROM
Port C						*			KEY	KEY	KEY	KEY	*	*	NAND	RESET	LAND	NAND		LAN		S3	*	S3			*	RDY	ADD	ADD	ADD	ADD

위 표의 블루칼라 부분은 해당 디바이스로 사용하지 않으면 모두 GPIO 포트 사용이 가능하며, 회색부분은 시스템에서 사용되므로 임의로 사용이 불가능하다.

구 분	내 용	GPIO 포트 수
S0 ~ S3	Serial Port 1 ~ 4	20
Debug	Debug Port	2
Reset	Reset	1
Rdy	Ready LED	1
ADC	Analog Digital Converter	4
LAN	LAN Port	2
EEPROM	SPI (EEPROM)	4
NAND	NAND Flash	2
KEY	Key Pad	8
*	GPIO & User Peripheral	12

Eddy-S4M 의 GPIO 구성표

bytes	3								2								1								0							
bits	3	3	2	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0		
bit	1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0
bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Port A		*								*																*						
Port B	*	*									*	*	*	*	*	*													*	*	*	*
Port C							*	*				*	*	*	*		*	*	*	*		*	*			*		*	*	A D C	A D C	

위 표의 블루칼라 부분은 해당 디바이스로 사용하지 않으면 모두 GPIO 포트 사용이 가능하며, 회색부분은 시스템에서 사용되므로 임의로 사용이 불가능하다.

구분	내용	GPIO 포트 수
ADC	Analog Digital Converter	2
*	GPIO & User Peripheral	32

Port A,B,C 각 포트는 32 개의 GPIO 를 표현할 수 있으므로 프로그램에서는 int 형 4바이트형 변수의 각 비트로 각각의 GPIO 포트를 표현한다.

```
struct eddy_gpio {
    Unsigned int value [3]; //Port A, B, C 각 해당 GPIO 채널에 대한 입출력 상태 값
    Unsigned int mode [3]; //Port A, B, C 각 해당 GPIO 채널에 대한 입력 또는 출력 설정
    Unsigned int pullup [3]; //Port A, B, C 각 해당 GPIO 채널에 대한 입력설정 시의
        //pullup 또는 pulldown 설정
    Unsigned int enable [3]; //Port A, B, C 각 해당 GPIO 채널에 대한 GPIO 사용 여부
};
```

enable 의 경우 비트가 0 → disable (GPIO 로 사용안함), 1 → Enable (GPIO 로 사용함)

mode 의 경우 비트가 0 → Input mode 로 설정,, 1 → Output mode 로 설정

value 의 경우 비트가 0 → 입력 또는 출력 상태가 Low, 1 → High

pullup 의 경우 비트가 0 → pulldown, 1 → pullup 상태

SETGPIOINIT

Function	Eddy 기동 직후 시스템의 GPIO 로 사용할 포트들을 초기화 한다.	
Format	void ioctl(int fd, SETGPIOINIT, struct *gpio_struct);	
Parameter	fd	GPIO 디바이스("/dev/eddy_gpio")를 오픈한 핸들번호
	gpio_struct	WEB 환경설정에서 등록한 GPIO 설정파일로 /etc/eddy_gpio.cfg 파일 상의 GPIO 테이블 값을 저장한 struct 포인터 <pre> struct gpio_struct { unsigned int value[3]; unsigned int mode[3]; unsigned int pullup[3]; unsigned int enable[3]; }; </pre>
Returns	None	
Notice	<p>Eddy-CPU 가 제공하는 GPIO 는 최대 56 포트이며, Eddy-S4M- 가 제공하는 GPIO 는 최대 34 포트이다.</p> <p>Eddy-CPU 의 경우 56 포트는 WAN 만을 사용했을 경우이며, 시리얼포트, ADC, Rese, RDY LED 등의 디바이스를 사용하는 경우에는 사용할 수 있는 GPIO 포트 수가 줄어 든다.</p> <p>Eddy-S4M 의 경우 34 포트는 공용으로 사용하는 디바이스가 ADC 하나이다.</p> <p>이 명령은 Eddy 가 부팅 직후 Pinetd.c 에서 환경설정에 등록된 디바이스를 제외한 나머지 포트를 GPIO 로 사용가능하도록 초기화하므로 , 사용자는 이 명령을 사용할 필요가 없으며, 불가피한 사용에는 각별히 주의할 필요가 있다.</p> <p>예를들어 시리얼포트를 사용하도록 WEB 환경설정을 하고 Eddy 를 재 부팅 하면 해당 포트는 GPIO 포트가 아닌 시리얼포트로 동작하게 되는데, 이 명령을 통해 강제로 시리얼포트를 GPIO 로 사용하도록 변경하면 시리얼 포트를 사용하는 어플리케이션이 오동작 하게 된다.</p>	

SETGPIOMOD_LM

Function	Port A, B, C 에 대한 입출력 방향을 일괄적으로 설정한다.
Format	void ioctl (int fd, SETGPIOMOD_LM, int *mode[3]);
Parameter	fd GPIO 디바이스("/dev/eddy_gpio")를 오픈한 핸들번호 mode Port A.B.C 의 mode 값을 저장한 버퍼 포인터 Bit 값이 0 이면 Input, 1 이면 Output
Returns	None
Notice	GPIO 로 사용하도록 설정한 bit 이외에 다른 Bit 에는 어떤 값으로 설정 하여도 관계 없음

GETGPIOMOD_LM

Function	Port A, B, C 에 대한 입출력 방향을 일괄적으로 읽어온다.
Format	void ioctl (int fd, GETGPIOMOD_LM, int *mode[3]);
Parameter	fd GPIO 디바이스("/dev/eddy_gpio")를 오픈한 핸들번호 mode Port A.B.C 의 mode 값이 저장될 버퍼 포인터
Returns	None
Notice	

SETGPIOVAL_LM

Function	Port A, B, C 의 Mode 가 모두 출력인 경우 출력값을 일괄 설정한다.
Format	void ioctl (int fd, SETGPIOVAL_LM, int *value[3]);
Parameter	fd GPIO 디바이스("/dev/eddy_gpio")를 오픈한 핸들번호 value Port A.B.C 의 value 값을 저장한 버퍼 포인터 Bit 값이 0 이면 Low, 1 이면 High
Returns	None
Notice	GPIO 로 사용하도록 설정한 bit 이외에 다른 Bit 에는 어떤 값으로 설정 하여도 관계 없음

GETGPIOVAL_LM

Function	Port A, B, C 의 입출력의 상태값을 일괄적으로 읽어온다.	
Format	void ioctl (int fd, GETGPIOVAL_LM, int *value[3]);	
Parameter	fd	GPIO 디바이스("/dev/eddy_gpio")를 오픈한 핸들번호
	value	Port A.B.C 의 value 값을 저장할 버퍼 포인터
Returns	None	
Notice		

SETGPIOPUL_LM

Function	Port A, B, C 의 Mode 가 모두 입력인 경우 pullup 상태를 일괄 설정한다.	
Format	void ioctl (int fd, SETGPIOPUL_LM, int *pullup[3]);	
Parameter	fd	GPIO 디바이스("/dev/eddy_gpio")를 오픈한 핸들번호
	pullup	Port A.B.C 의 pullup 값을 저장한 버퍼 포인터 Bit 값이 0 이면 Pulldown, 1 이면 Pullup
Returns	None	
Notice	GPIO 로 사용하도록 설정한 bit 이외에 다른 Bit 에는 어떤값으로 설정 하여도 관계 없음	

GETGPIOPUL_LM

Function	Port A, B, C 의 pullup 상태를 일괄 읽어온다.	
Format	void ioctl (int fd, GETGPIOPUL_LM, int *pullup[3]);	
Parameter	fd	GPIO 디바이스("/dev/eddy_gpio")를 오픈한 핸들번호
	pullup	Port A.B.C 의 pullup 값을 저장할 버퍼 포인터
Returns	None	
Notice		

SETGPIOMOD_LA
SETGPIOMOD_LB
SETGPIOMOD_LC

Function	Port A, B, C 중 하나의 포트에 대해 입출력 방향을 설정한다.	
Format	void ioctl (int fd, SETGPIOMOD_L?, int *mode);	
Parameter	fd	GPIO 디바이스("/dev/eddy_gpio")를 오픈한 핸들번호
	mode	Port 의 mode 값을 저장한 버퍼 포인터
		Bit 값이 0 이면 Input, 1 이면 Output
Returns	None	
Notice	GPIO 로 사용하도록 설정한 bit 이외에 다른 Bit 에는 어떤값으로 설정 하여도 관계 없음	

GETGPIOMOD_LA
GETGPIOMOD_LB
GETGPIOMOD_LC

Function	Port A, B, C 중 하나의 포트에 대해 입출력 방향의 상태를 읽어온다.	
Format	void ioctl (int fd, GETGPIOMOD_L?, int *mode);	
Parameter	fd	GPIO 디바이스("/dev/eddy_gpio")를 오픈한 핸들번호
	mode	Port 의 mode 값을 저장할 버퍼 포인터
Returns	None	
Notice		

SETGPIOVAL_LA
SETGPIOVAL_LB
SETGPIOVAL_LC

Function	출력으로 설정된 포트인 경우 포트의 출력값을 설정한다.	
Format	void ioctl (int fd, SETGPIOVAL_L?, int *value);	
Parameter	fd	GPIO 디바이스("/dev/eddy_gpio")를 오픈한 핸들번호
	Value	Port 의 mode 값을 저장한 버퍼 포인터
		Bit 값이 0 이면 Low, 1 이면 High
Returns	None	
Notice	GPIO 로 사용하도록 설정한 bit 이외에 다른 Bit 에는 어떤값으로 설정 하여도 관계 없음	

GETGPIOVAL_LA
GETGPIOVAL_LB
GETGPIOVAL_LC

Function Port A, B, C 중 하나의 포트에 대해 입출력 상태를 읽어온다.

Format void ioctl (int fd, GETGPIOVAL_L?, int *value);

Parameter fd GPIO 디바이스("/dev/eddy_gpio")를 오픈한 핸들번호
 value Port 의 mode 값을 저장할 버퍼 포인터

Returns None

Notice

SETGPIOPUL_LA
SETGPIOPUL_LB
SETGPIOPUL_LC

Function 입력으로 설정된 Port 의 Pullup 상태를 설정한다.

Format void ioctl (int fd, SETGPIOPUL_L?, int *pullup);

Parameter fd GPIO 디바이스("/dev/eddy_gpio")를 오픈한 핸들번호
 pullup Port 의 pullup 값을 저장한 버퍼 포인터
 Bit 값이 0 이면 Pulldown, 1 이면 Pullup

Returns None

Notice GPIO 로 사용하도록 설정한 bit 이외에 다른 Bit 에는 어떤 값으로 설정 하
 여도 관계 없음

GETGPIOPUL_LA
GETGPIOPUL_LB
GETGPIOPUL_LC

Function Port A, B, C 중 하나의 포트에 대해 pullup 상태를 읽어온다.

Format void ioctl (int fd, GETGPIOPUL_L?, int *pullup);

Parameter fd GPIO 디바이스("/dev/eddy_gpio")를 오픈한 핸들번호
 pullup Port 의 pullup 값을 저장할 버퍼 포인터

Returns None

Notice

6.8 ADC 관련 함수

Eddy-CPU 는 ADC (Analog Digital Converter) 4 채널을 제공한다. Eddy DK 보드에는 온도센서와 조도센서가 시험

용도로 설치되어 있어 이를 통해 ADC 를 통해 센서의 상태정보를 실시간으로 확인할 수 있다. ADC 인터페이스를 이용하는 샘플 프로그램은 Eddy_APPS/test_adc.c 이므로 사용자는 이 소스를 응용하여 프로그램 개발이 가능하다.

ADCSETCHANNEL

Function	ADC 디바이스의 4 개 채널의 사용여부를 설정한다.		
Format	void ioctl(int fd, ADCSETCHANNEL, int *channel);		
Parameter	fd	ADC 디바이스("/dev/adc")를 오픈한 핸들번호	
	channel	채널의 사용여부를 등록한 버퍼 포인터	
Returns			
Notice	x x x x x x x (bits)		
	-----	channel	1 (temperature sensor)
	-----	channel	2 (illumination sensor)
	-----	channel	3 (future use)
	-----	channel	4 (future use)

ADCGETVALUE

Function	ADC 디바이스의 4 개 채널의 동작상태 정보를 읽어온다.		
Format	void ioctl (int fd, ADCGETVALUE, struct adc_struct *channels);		
Parameter	fd	ADC 디바이스("/dev/adc")를 오픈한 핸들번호	
	channels	4 개 채널의 동작상태를 저장할 버퍼 포인터	
Returns			
Notice	struct adc_value {		
	int ch1_value;		
	int ch2_value;		
	int ch3_value;		
	int ch4_value;		
	};		

6.9 RTC 관련 함수

Eddy-CPU 는 DK 에 RTC (Real Time Clock) 를 별도로 제공한다. 프로그램에 의한 시간설정 이외에 Busybox 에 서 제공하는 Date 와 rdate 를 통해서도 RTC 에 날짜와 시간을 설정할 수 도 있다. RTC 디바이스를 사용하는 샘플 프로그램은 Eddy_APPS/test_rtc.c 이므로 사용자는 이 소스를 응용하여 프로그램 개발이 가능하다.

RTC_SET_TIME

Function	RTC 디바이스에 날짜와 시간을 설정한다.		
Format	void ioctl (int fd, RTC_SET_TIME, struct tm *tm);		
Parameter	fd	RTC 디바이스("/dev/rtc0")를 오픈한 핸들번호	
	tm	Linux 표준 time 관련 인터페이스를 위한 struct tm 과 호환하며, 설정할 시간이 저장된 struct 포인터	
Returns			
Notice	Linux 에서 제공하는 표준 time () 라이브러리와는 호환되지 않는다.		

RTC_RD_TIME

Function	RTC 디바이스로부터 날짜와 시간을 읽어온다.		
Format	void ioctl (int fd, RTC_RD_TIME, struct tm *tm);		
Parameter	fd	RTC 디바이스(“/dev/rtc0”)를 오픈한 핸들번호	
	tm	Linux 표준 time 관련 인터페이스를 위한 struct tm 과 호환하며, 읽어온 시간을 저장할 struct 포인터	
Returns			
Notice	Linux 에서 제공하는 표준 time () 라이브러리와는 호환되지 않는다. Mktime () 함수를 이용하여 time_t 형 으로 변환하여 사용할 수 있다.		

6.10 Debugging 관련 함수

Eddy 는 Telnet 을 통해 각 어플리케이션의 실행상태를 실시간으로 디버깅할 수 있다. 각 어플리케이션의 SB_DEBUG 가 ON 이 되면 Telnet 화면에 디버그 로그 메시지가 출력될 수 있도록 지원하는 함수이다.

SB_LogDataPrint

Function	데이터를 각 바이트별로 Hex 또는 Ascii code 로 Print 출력한다.
Format	void SB_LogDataPrint (char *RTx, char *buff, int data_len);
Parameter	<div>*RTx 데이터의 설명 메시지</div> <div>*Buff 출력할 데이터가 저장된 버퍼의 주소</div> <div>Data_len Data 의 크기</div>
Returns	None
Notice	<p>메시지를 첫번째 로그인 한 Telnet sh 에 출력한다.</p> <p>출력에는 Eddy 의 1msec 단위로 증가하는 시스템 Tick Counter 을 포함하여 다음의 형식으로 출력한다.</p> <pre>SB_LogDataPrint ("Send" , "\t12345\n" , 8);</pre> <pre>[191020202] Send 8 = 08,1,2,3,4,5,0d,0a</pre> <pre>-----</pre> <pre>Tick Counter RTx data_Len buff</pre> <p>Eddy 의 각 어플리케이션의 디버깅은 Def 명령을 통해 다음과 같이 설정할 수 있다. (def.c 참조)</p> <pre># def po <1/2/all> debug <on/off></pre>

SB_LogMsgPrint

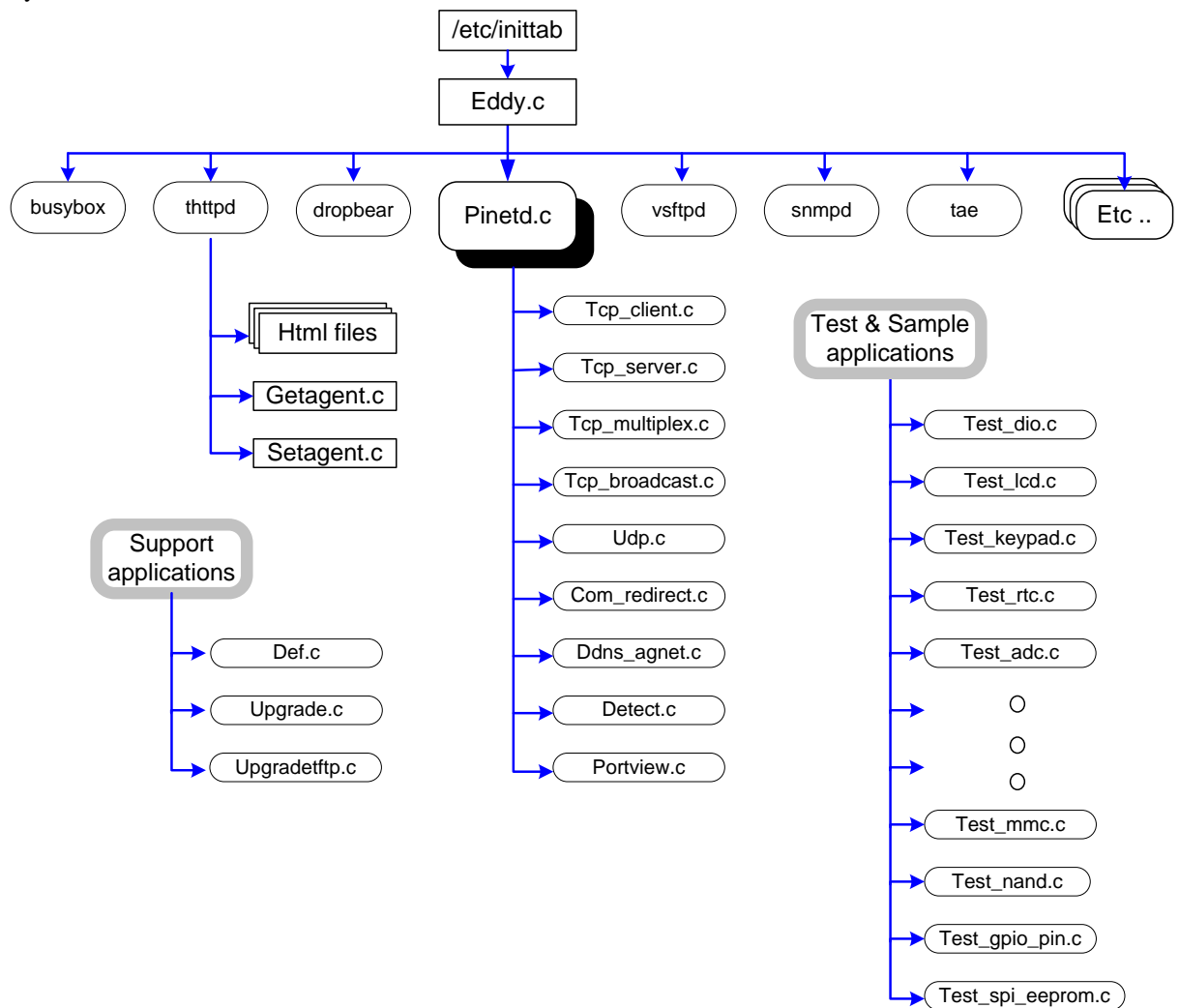
Function	Printf 와 같은 형식의 포맷으로 출력한다.
Format	void SB_LogMsgPrint (const char *Format, ...);
Parameter	*Format Printf 의 형식포맷
Returns	None
Notice	<p>메시지를 첫번째 로그인 한 Telnet sh 에 출력한다.</p> <p>출력에는 Eddy 의 1msec 단위로 증가하는 시스템 Tick Counter 을 포함하여 다음의 형식으로 출력한다.</p> <pre>SB_LogMsgPrint ("%s means Real-Time\n" , "Eddy");</pre> <pre>[191020202] Eddy means Real-Tile</pre> <p>Eddy 의 각 어플리케이션의 디버깅은 Def 명령을 통해 다음과 같이 설정할 수 있다. (def.c 참조)</p>

7장. Eddy Software

본 절에서는 Eddy DK 에 포팅된 software 구조에 대한 설명이다. Eddy DK 는 Com_redirect, gdbserver, tae, SB_APIs 라이브러리 를 제외한 모든 소스로 공개하므로 DK 개발자는 제공되는 소스를 이용하여 원하는 기능을 쉽게 구현하여 펌웨어를 개발할 수 있다.

7.1 Software 구성도

Eddy 는 부팅 후 처음 실행되는 eddy.c 를 시작으로 web 또는 def.c 에서 설정한 환경설정 정보를 읽어 실행된다. Eddy 가 제공하는 모든 어플리케이션은 모두 6장의 라이브러리를 통해 개발되어 있으므로 참고하기 바란다.



7.2 주요 Application 설명

Eddy 의 가장 중요한 역할을 하는 eddy.c 와 pinetd.c 를 설명한다. 이 외의 어플리케이션은 pinetd.c 에서 실행하고, 감시되는 어플리케이션들과 사용자에게 의해 수동으로 실행되는 애플리케이션으로 구분된다. 각 어플리케이션의 간단한 기능설명은 “4.1 프로그램 종류” 를 참고한다.

7.2.1 eddy.c

Eddy 부팅 직후 실행되는 프로그램으로, /flash 에 저장된 환경 파일을 읽어온다. 환경설정 정보로 네트워크를 초기화 하고, 각종 데몬 프로그램을 실행한다. 환경파일이 /flash 에 존재하지 않으면 환경설정 파일을 공장초기화 값으로 생성한다.

7.2.2 pinetd.c

Eddy.c 에 의해 실행되는 eddy 최상위 데몬 프로그램으로 하위 프로세서를 실행하고 감시한다. 주기적으로 Eddy 의 동작상태를 Ready Led 를 통해 확인할 수 있고 점멸한다. 리셋 스위치를 주기적으로 감시하여 사용자의 리셋 요구나 공장초기화 요구가 있는지 감시한다. /flash 의 환경 설정 파일을 이후 각 어플리케이션에서 참조 가능하도록 /etc 폴더에 복사한다.

7.2.3 그밖의 어플리케이션

환경파일에서 정의된 각 시리얼포트의 프로토콜에 따라 실행되는 어플리케이션은 다음과 같다.

Tcp_server, tcp_client, com_redirect, tcp_broadcast, tcp_multiplex, udp (udp_server/client)

시리얼포트와는 관계없이 외부 네트워크의 서비스 기능을 수행하는 어플리케이션은 다음과 같다.

Portview, detect, ddns_agent

telnet 접속으로 사용자가 수동으로 실행할 수 있는 어플리케이션은 다음과 같다.

Def, upgrade, upgradetftp

Eddy DK v2.1 보드의 디바이스를 시험하기 위한 어플리케이션은 다음과 같다.

test_sio, test_dio, test_lcd, test_keypad, test_spi_eeprom, test_nand, test_sd, Test_adc, test_gpio_pin, test_gpio_led, testdk

소켓, 시리얼포트 응용 프로그램을 작성하기 위한 샘플 소스는 다음과 같다.

Test_serial, test_serial_to_lan-1, test_serial_lan-2, test_tcp_server, test_tcp_client, test_udp_server, test_udp_clinet, test_read_config, test_bluetooth

8장. HTML 및 CGI 변경하기

본 장에서는 환경 설정의 용도로 사용되는 HTML 파일 및 HTML 코드가 호출하는 CGI 모듈에 대해 설명한다. 제공되는 CGI 소스 및 HTML 문서는 실제 Eddy DK 가 사용중인 파일들 이므로 적절히 수정하여 사용자가 원하는 방식으로 수정할 수 있다.

8.1 WEB Configuration

Eddy 에서 실행되는 전체소스는 src/Eddy_APPS/web/htdocs 에 위치하며, Html 에서 필요한 정보를 지원하는 CGI 소스는 src/Eddy_APPS/web/cgi 에 위치한다.

getagent.c

/etc 폴더에 있는 임시 환경설정파일을 읽어들이어 해당 HTML 페이지에 설정 값을 전달하여 브라우저를 통해 설정정보를 보여줄 수 있도록 하는 역할을 한다.

setagent.c

HTML 페이지 상에서 사용자가 수정한 환경설정 내용을 읽어 들여 /etc 폴더에 있는 임시 환경설정파일에 다시 저장하는 역할을 수행하는 프로그램이다. 최종으로 Save & Reboot 시에는 환경설정 파일은 /flash 에 저장된다.

8.2 예제 HTML 코드

아래 예는 main.html 소스 중 일부이다. 아래의 예와 같이 HTML 에서는 C 언어에서처럼 변수를 사용하여 코딩을 작성할 수 없기 때문에 CGI 와 연계되는 심볼을 통해 값을 넘겨받아 처리한다. 아래에서처럼 빨간색으로 표시한 부분은 getagent.c 에서 값을 넘겨주는 심볼 링크이다.

(network.html 소스 요약)

```
<tr bgcolor="#FFFFFF">
<td class="content">IP Address</td>
<td class="content"><input type="text" size="16" maxlength="16" name="N_IP" value="[v,n_ip]" >

<tr bgcolor="#FFFFFF">
<td class="content">Subnet Mask</td>
<td class="content"><input type="text" size="16" maxlength="16" name="N_MASK" value="[v,n_mask]" >

<tr bgcolor="#FFFFFF">
<td class="content">Gateway</td>
<td class="content"><input type="text" size="16" maxlength="16" name="N_GW" value="[v,n_gw]" >

<tr bgcolor="#FFFFFF">
<td class="content">DNS</td>
<td class="content"><input type="text" size="16" maxlength="16" name="N_DNS" value="[v,n_dns]" >

<tr bgcolor="#FFFFFF">
<td class="content">Telnet Service</td>
<td class="content"><select name="N_TELNET">
```

```
<option [v, n_telnet_di] value="0">Disable</option>
<option [v, n_telnet_en] value="1">Enable</option>
</select>

<tr bgcolor="#FFFFFF">
<td class="content">Telnet Service</td>
<td class="content"><select name="N_WEB">
<option [v, n_web_di] value="0">Disable</option>
<option [v, n_web_en] value="1">Enable</option>
</select>
```

위의 경우에서처럼 CGI 와 연계를 위해 각 레코드별로 name 과 value 부분이 있는데, name 은 HTHM 에서 사용자가 수정한 정보를 보관하여 HTML 페이지의 하단에 submit 버튼을 클릭 시 setagent.c 를 통해 수정한 정보를 저장하는 역할을 하며, value 는 getagent.c 로 값을 읽어와 HTML 페이지에 표시하고 사용자가 원하는 값으로 수정할 수 있도록한다.

8.3 예제 CGI 코드

Eddy DK 의 cgi 파일은 getagent.cgi 와 setagent.cgi 로 구성되어 있으며, getagent.c 는 HTML 문서에 /etc/ 폴더에 환경파일을 읽어 환경설정 정보를 보여 줄 수 있도록 하며, setagent.c 는 HTML 문서에서 사용자가 수정한 정보를 다시 /etc 폴더에 환경파일을 저장하는 역할을 하고 최종적으로는 /flash 에 저장하여 Eddy 를 리셋하여도 사용자가 설정한 환경설정 정보를 기억하도록 한다.

아래는 위의 예제 HTML 페이지에 설정값을 보여주기 위한 getagent.c 의 처리 부분이다.

[소스 요약]

```
if (cgiFormStringNoNewlines("N_IP", buff, 16) == cgiFormNotFound) {
    sprintf(buff, "%d.%d.%d.%d", cfg.system.ip[0], cfg.system.ip[1], cfg.system.ip[2], cfg.system.ip[3]);
    listPutf(list, "n_ip", buff);
}
else
    listPutf(list, "n_ip", buff);

if (cgiFormStringNoNewlines("N_MASK", buff, 16) == cgiFormNotFound) {
    sprintf(buff, "%d.%d.%d.%d", cfg.system.mask[0], cfg.system.mask[1],
    cfg.system.mask[2], cfg.system.mask[3]);
    listPutf(list, "n_mask", buff);
}
Else
    listPutf(list, "n_mask", buff);

if (cgiFormStringNoNewlines("N_GW", buff, 16) == cgiFormNotFound) {
    sprintf(buff, "%d.%d.%d.%d", cfg.system.gateway[0], cfg.system.gateway[1],
    cfg.system.gateway[2], cfg.system.gateway[3]);
    listPutf(list, "n_gw", buff);
}
Else
    listPutf(list, "n_gw", buff);

if (cgiFormStringNoNewlines("N_DNS", buff, 16) == cgiFormNotFound) {
    sprintf(buff, "%d.%d.%d.%d", cfg.system.dns[0], cfg.system.dns[1],
    cfg.system.dns[2], cfg.system.dns[3]);
    listPutf(list, "n_dns", buff);
}
else
    listPutf(list, "n_dns", buff);

cgiFormInteger("N_TELNET", &value, cfg.system.telnet_server);
if (value == 1) {
    listPutf(list, "n_telnet_di", "");
    listPutf(list, "n_telnet_en", "selected");
}
else {
    listPutf(list, "n_telnet_di", "selected");
    listPutf(list, "n_telnet_en", "");
}

cgiFormInteger("N_WEB", &value, cfg.system.web_server);
if (value == 1) {
    listPutf(list, "n_web_di", "");
    listPutf(list, "n_web_en", "selected");
} else {
    listPutf(list, "n_web_di", "selected");
    listPutf(list, "n_web_en", "");
}
}
```

아래는 HTML 페이지에서 “submit” 버튼을 클릭했을 때 사용자가 변경한 설정 값을 저장하기 위한 setagent.c 의 처리 부분이다.

[소스 요약]

```
value2 = cgiFormStringNoNewlines("N_IP", buff, 16);
if (value2 != cgiFormEmpty) convert_address (buff, cfg.system.ip);

value2 = cgiFormStringNoNewlines("N_MASK", buff, 16);
if (value2 != cgiFormEmpty) convert_address (buff, cfg.system.mask);

value2 = cgiFormStringNoNewlines("N_GW", buff, 16);
if (value2 != cgiFormEmpty) convert_address (buff, cfg.system.gateway);

value2 = cgiFormStringNoNewlines("N_DNS", buff, 16);
if (value2 != cgiFormEmpty) convert_address (buff, cfg.system.dns);

cgiFormInteger("N_TELNET", &value, cfg.system.telnet_server);
cfg.system.telnet_server = value;

cgiFormInteger("N_WEB", &value, cfg.system.web_server);
cfg.system.web_server = value;
```

9장. 부록

본 장에서는 Eddy 의 Flash 가 손상되어 부팅이 되지 않는 경우에 복구하는 방법을 설명한다.

9.1 부트로더로 시스템 복구

사용자 영역의 Flash 가 손상되더라도 시스템 부팅에 영향을 미치지 않는다. 그러나 사용자 프로그램 오류로 인해 시스템이 지속적으로 재 부팅하거나 잘못된 IP 설정으로 인한 장비 접속이 불가할 시에는 Eddy 을 공장 출하 상태로 만들어야 한다.

공장 출하 상태로 돌리기 위해서는 Boot Loader 에서 펌웨어를 다시 올리는 방법이 있다. 이 방법을 위해서는 리눅스 환경을 구축한 컴퓨터에 TFTP server 가 설치가 되어 있어야 한다.

주의:

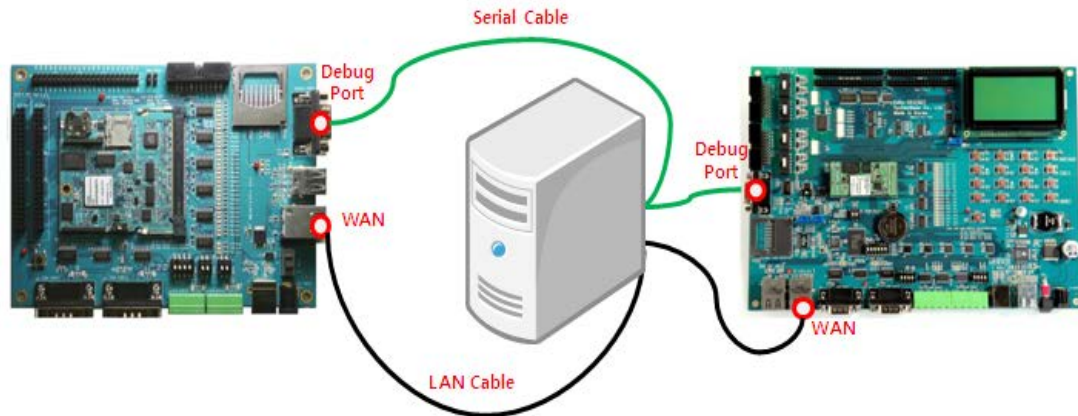
bootloader 가 손상된 경우 복구가 불가능하다. 따라서 bootloader 에서는 매뉴얼에서 제공한 명령어 외의 명령어를 사용하지 않도록 한다.

9.1.1 리눅스 환경에서 TFTP 설치

다음은 bootloader 에서 System 을 복구하는 방법을 Fedora core 5 운영체제를 기반으로 설명하고 있다. 다른 운영체제를 사용하고 있다면 그에 맞는 tftp-server 와 xinetd 데몬이 필요하다. Tftp-server 를 설치하기 위해서는 xinetd 데몬이 필요하다. 먼저 tftp-server 와 xinetd 데몬이 설치가 되어있는 것에 대해서 확인을 한 후 설치가 되어 있지 않으면 설치를 한다. 설치 후에는 제공된 SDK의 firmware폴더에서 만들어진 firmware를 tftp의 기본 폴더로 이동한다. Fedora Core 5에서는 /tftpboot 폴더가 기본 폴더다..

9.1.2 하드웨어 설치 및 복구

제공한 LAN Cable을 통해 컴퓨터의 LAN 포트와 DK 보드의 WAN 포트를 연결한다. 제공된 Serial Cross Cable을 통해 디버그 포트와 컴퓨터의 시리얼 포트를 연결한 후 minicom을 통해서 컴퓨터의 시리얼 포트에 접속한다. 컴퓨터의 시리얼 포트 설정은 115200 bps, 8 data bit, No parity, 1 stop bit로 설정한 후 DK보드의 전원을 인가한다.



[Eddy-S4M DK인 경우]

[Eddy-CPU DK인 경우]

인가 후에 다음과 같은 메시지가 minicom을 통해 출력된다. 출력이 시작되면 엔터를 쳐서 bootloader 로 진입한다. 아래는 bootloader로 진입한 후의 이미지 이다.

```
NAND: 256 MB
Macb0: Autonegotiation complete
Macb0: link up, 100 Mbps full-duplex (lpa: 0x45e1)
Hit any key to stop autoboot: 0
U-Boot>
U-Boot>
```

Bootloader 에서는 커널, 펌웨어 이미지를 명령어를 통해 Flash 메모리에 복사하여 복구가 가능하다. 커널과 펌웨어 이미지 파일을 업그레이드 하기 위해서는 부트 로더에 Eddy 의 가상 IP 주소와 TFTP 서버의 IP 주소를 반드시 설정 해 주어야 한다. 현재 설정을 확인하려면 “printenv” 명령을 통해 현재 부트로더에 설정된 Eddy 와 TFTP 서버의 IP 주소를 확인 할 수 있다.

```

U-Boot> printenv

.
.

ethaddr=00:05:F4:11:22:33
Config_Size=10000
stdin=serial
stdout=serial
stderr=serial
OS_Size==20000000
filesize=1f0f07
fileaddr=20000000
netmask=255.255.255.0
ipaddr=192.168.0.223      ← Eddy 의 IP Address
serverip=192.168.0.220   ← TFTP 서버의 IP Address
FileSystem_Size=0
.
    
```

위의 이미지 중 붉은 색으로 표시된 부분이 펌웨어 버전 정보이다. 이 두 버전 정보가 일치 하지 않는다면 다음의 과정을 통해서 일치 시킨다.

Eddy 의 임시 IP 주소와 TFTP 서버의 IP 주소를 변경하려면 다음과 같이 실행한다.

```

U-Boot> setenv serverip <TFTP 서버의 IP 주소>
U-Boot> setenv ipaddr   <Eddy 의 임시 IP 주소>
U-Boot> saveenv
    
```

IP 정보가 확인 되었으면 복구를 시작한다.

install bootloader <부트로더 펌웨어 이름> ; 부트로더영역을 복구

(주의: 부트로더가 손상되면 DK 보드로는 복구가 불가능)

install os <OS 펌웨어 이름> ; OS 영역을 복구

install fs <File System 펌웨어 이름> ; File System 영역을 복구

다음과 같이 실행하면 설정된 TFTP 서버로부터 이미지 파일을 다운로드 하여 복구한다.

다음은 OS 영역을 복구하는 과정이다.

```
U-Boot> install os eddy-os-2.1.x.x.bin
TFTP from server 192.168.0.220; our IP address is 192.168.0.223
Filename 'eddy-os-2.1.x.x.bin'.
Load address: 0x20000000
Loading:#####
#####
done
Bytes transferred = 1112284 (10f8dc hex)
.
.
U-Boot>
```

다음은 File System 영역을 복구하는 과정이다.

```
U-Boot> install fs eddy-fs-2.1.x.x.bin
TFTP from server 192.168.0.220; our IP address is 192.168.0.223
Filename 'eddy-fs-2.1.x.x.bin'.
Load address: 0x20000000
Loading:#####
#####
#####
#####done
Bytes transferred = 2035463 (1f0f07 hex)
.
.
U-Boot>
```

복구가 완료 되었으면 “boot” 명령을 통해 부팅을 시도한다.

```
U-Boot> boot
```


9.1.3 복구 시 문제점 해결

```
U-Boot> install os eddy-os-21.1.x.x.bin
TFTP from server 192.168.0.220; our IP address is 192.168.0.223
Filename 'eddy-os-21.1.x.x.bin'.
Load address: 0x20000000
Loading: .....
```

위와 같은 메시지가 출력되고 진행이 되지 않을 경우 WAN의 연결 상태를 확인하고 tftp-server가 설치된 PC의 IP가 192.168.0.220으로 설정되어 있는지 확인한다.(위의 예시 기준으로 설명)

```
U-Boot> install fs eddy-fs-2.1.x.x.bin
TFTP from server 192.168.0.220; our IP address is 192.168.0.223
Filename 'eddy-fs-2.1.x.x.bin'.
Load address: 0x20000000
Loading
TFTP error: 'File not found' (1)
Starting again
```

위의 붉은 색으로 표시된 이름이 tftp-server가 설치된 PC의 펌웨어 이름과 동일 하여야 한다.

```
U-Boot> install os eddy-os-21.x.x.bin
TFTP from server 192.168.0.220; our IP address is 192.168.0.223
Filename 'eddy-os-2.1.x.x.bin'.
Load address: 0x20000000
Loading: TTTT#TTT#
```

위와 같은 메시지가 출력되는 경우는 네트워크 상에서 같은 MAC address 를 가지고 있거나 같은 IP를 가지고 있는 제품이 있는 경우이다. 이 경우 같은 네트워크 상에 같은 eddy 제품이 있는지 확인을 한다.

9.2 USB 포트로 시스템 복구

사용자 영역의 Flash 가 손상되더라도 시스템 부팅에 영향을 미치지 않는다. 그러나 사용자 프로그램 오류로 인해 시스템이 지속적으로 재 부팅되거나 잘못된 IP 설정으로 인한 장비 접속이 불가할 시에는 Eddy 를 공장 출하 상태로 만들어야 한다.

본 장은 USB 를 통하여 펌웨어를 다시 올림으로써 공장 출하 상태로 돌리는 방법을 설명하고자 한다.

(주의: USB 시스템 복구는 사용자 PC의 USB 포트 특성에 영향을 받을 수 있습니다.)

9.2.1 USB 시스템 복구 준비

Eddy-CPU v2.1과 Eddy-CPU v2.5 제품에 따라 설치 방법이 다르니, 이 점을 유의하여 설치 한다.

그리고, Eddy-S4M v2.1은 Eddy-CPU v2.1과 동일하며 Eddy-S4M v2.5는 Eddy-CPU v2.5와 동일하다.

Eddy-CPU/mp v2.5는 Eddy-CPU v2.5와 동일하다.

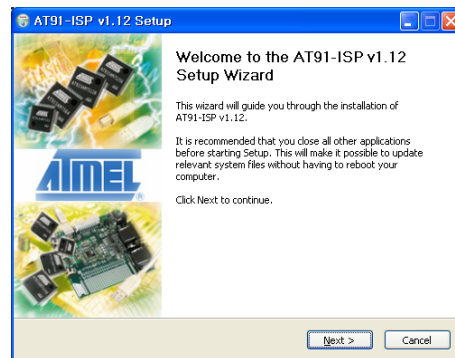
Eddy-CPU v2.1 또는 Eddy-S4M v2.1 인 경우

Eddy DK CD 안에 SDK\Windows\USB_recovery 폴더에서 Eddy-CPU_v21_USB_Recovery.zip 압축파일을 임의의 폴더 (예를 들면 C:\SystemBase\USB_recovery) 를 생성하여 압축을 푼다.

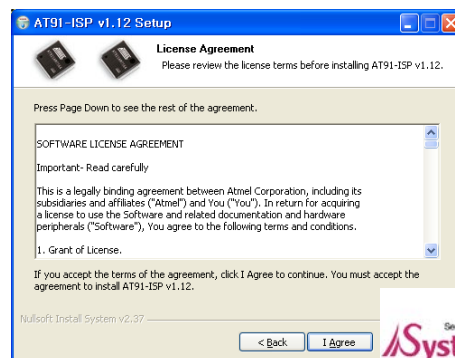
- USB Tool 프로그램을 설치 한다.

AT91-ISP.exe 파일을 더블 클릭함으로써
프로그램 설치를 시작한다.

Next 를 선택한다.

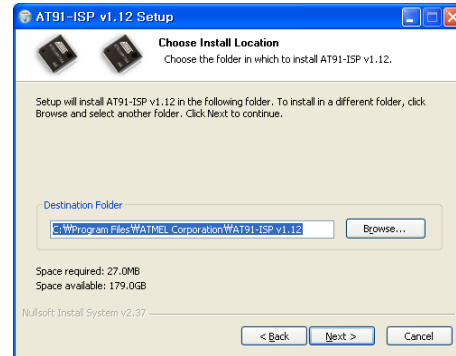


- I Agree 를 선택한다.

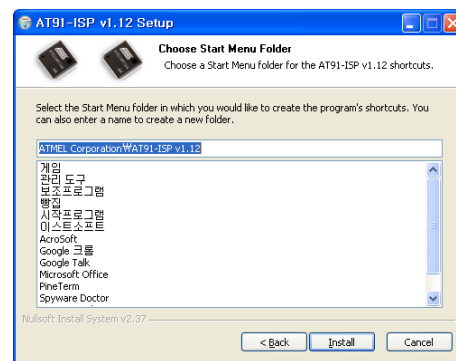


- 아래와 같이 폴더를 지정한 후 Next 를 선택한다.

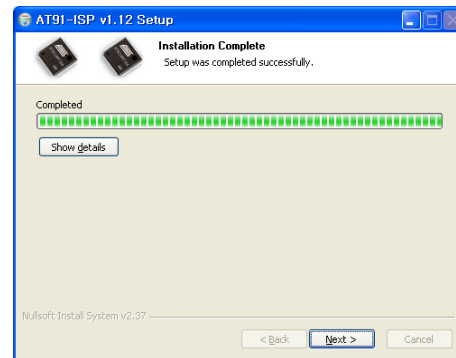
C:\ProgramFiles\ATMELCorporation\
AT91-ISP v1.12



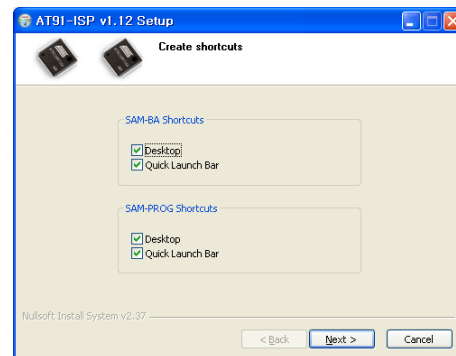
- Install 를 선택한다.



- Next 를 선택한다.



- 단축아이콘을 생성하고자 한다면 Desktop 또는 Quick Launch Bar 에 체크하고 Next 를 선택한다.



- Reboot now 를 선택한 후 Finish 를 선택한다.

시스템이 다시 부팅되면 CD 에서 제공되는
isp-extram-at91sam9260.bin 파일을 아래
폴더에 복사한다.

C:\ProgramFiles\ATMELCorporation\AT91-ISP
v1.12\SAM-BA v2.8\lib\AT91SAM9260-EK
프로그램 설치가 종료되면 Eddy DK v2.1
보드 드라이버 설치를 준비한다.



- 압축이 풀린 파일 중에 isp-extram-at91sam9260.bin 파일을 USB Tool 프로그램 설치 후에 다음 경로에
C:\ProgramFiles\ATMELCorporation\AT91-ISP v1.12\SAM-BA v2.8\lib\AT91SAM9260-EK 에 덮어쓰기 한다.

- 압축을 푼 폴더 안에 firmware 폴더에 있는 다음과 같은 펌웨어 파일을 확인한다.

eddy-bl-2.1.x.x.bin (Boot Loader),
eddy-bs-2.1.x.x.bin (Boot Strap File Name),
eddy-os-2.1.x.x.bin (Kernel File Name),
eddy-fs-2.1.x.x.bin (File System File Name)

- Eddy_burning_DataFlash.bat: 이 파일은 TCL 파일을 실행하여 USB를 통하여 펌웨어를 DK 보드에 작성하고 업그
레이드 후 로그 파일을 생성하는 역할을 한다. 반드시 아래와 같이 이 파일 내용 안에 eddy-bl-2.1.x.x.bin 파일
이름과 Eddy_burning_DataFlash.tcl 파일 이름이 다운로드 한 파일 이름과 동일하게 작성되어 있는지 확인한다.

```
sam-ba.exe \usb\ARM0 AT91SAM9260-EK Eddy_burning_DataFlash.tcl / eddy-bl-2.1.x.x.bin >
logfile.log
```

- Eddy_burning_DataFlash.tcl: 펌웨어 각각의 파일을 보드에 쓰는 역할을 한다. 이 TCL 파일도 마찬가지로 아래와
같이 eddy-bs-2.1.x.x.bin, eddy-os-2.1.x.x.bin, 그리고 eddy-fs-2.1.x.x.bin 파일 이름이 다운로드 한 파일 이름과
동일한지 확인하여야 한다.

```
...

#####

# Main script: Load the linux demo in DataFlash,
#
#           Update the environment variables
#####

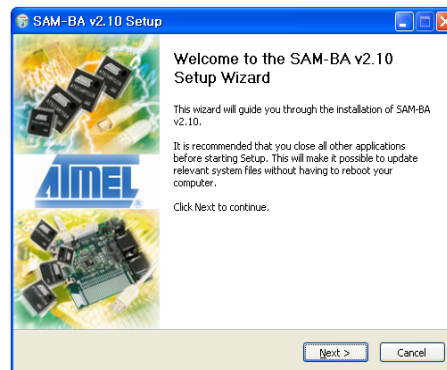
array set df_mapping {
    bootstrapFileName "eddy-bs-2.1.X.X.bin"
    kernelFileName    "eddy-os-2.1.X.X.bin"
    filesystemFileName "eddy-fs-2.1.X.X.bin"
}
```

Eddy-CPU v2.5, Eddy-CPU/mp v2.5 또는 Eddy-S4M v2.5 인 경우

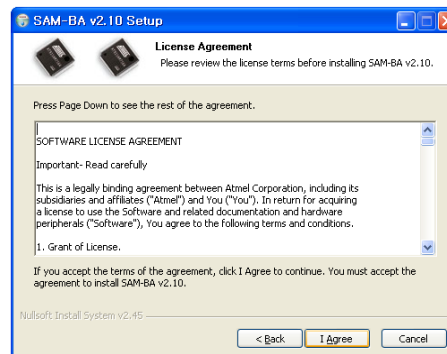
Eddy DK CD 안에 SDK\Windows\USB_recovery 폴더에서 Eddy-CPU_v25_USB_Recovery.zip 압축파일을
임의의 폴더 (예를 들면 C:\SystemBase\USB_recovery) 를 생성하여 압축을 풀다.

- USB Tool 프로그램을 설치 한다.

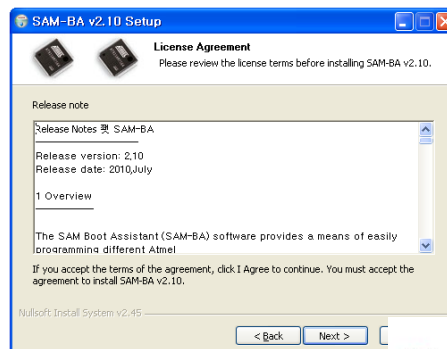
Sam-ba_2.10.exe 파일을 더블 클릭함으로써
프로그램 설치를 시작한다.
Next 를 선택한다.



- I Agree 를 선택한다.

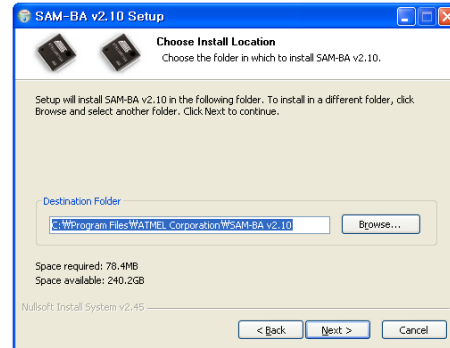


- Next 를 선택한다.

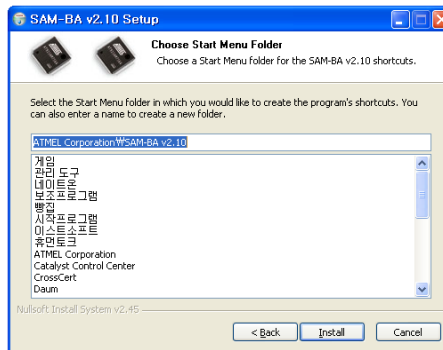


- 아래와 같이 폴더를 지정한 후 Next 를 선택한다.

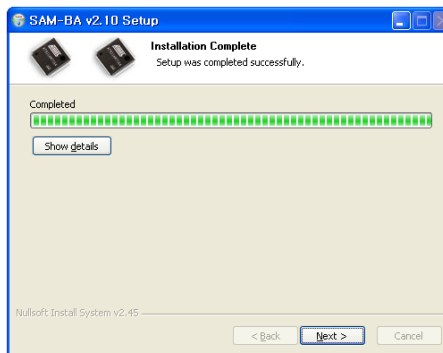
C:\ProgramFiles\ATMELCorporation\
SAM-BA v2.10



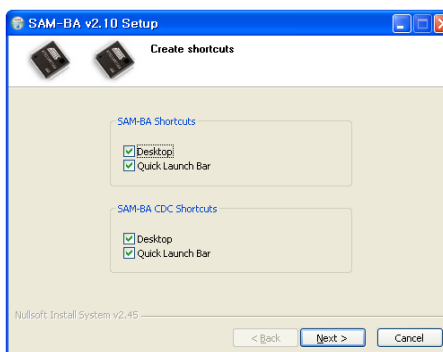
- Install 를 선택한다.



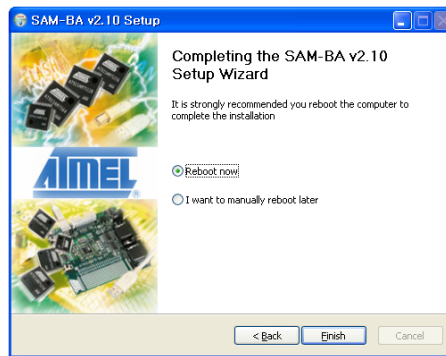
- Next 를 선택한다.



- 단축아이콘을 생성하고 싶으면 Desktop 또는 Quick Launch Bar 에 체크하고 Next 를 선택한다.



- Reboot now 를 선택한 후 Finish 를 선택한다.



- 압축이 풀린 파일 중에 at91sam9g20-ek.tcl 과 isp-serialflash-at91sam9g20.bin 파일은 다음 경로에 C:\Program Files\ATMEL Corporation\SAM-BA v2.10\tcl_lib\at91sam9g20-ek 에 덮어 쓰기 한다.
- 압축을 푼 폴더 안에 firmware 폴더에 있는 다음과 같은 펌웨어 파일을 확인한다.
(Eddy-CPU/mp v2.5 32bit는 firmware /32bit only 폴더의 펌웨어를 이용해야 한다.)

eddy-bl-2.5.x.x.bin (Boot Loader)

eddy-bs-2.5.x.x.bin (Boot Strap File Name)

eddy-os-2.5.x.x.bin (Kernel File Name)

eddy-fs-2.5.x.x.bin (File System File Name)

- Eddy_burning_SerialFlash.bat: 이 파일은 TCL 파일을 실행하여 USB를 통하여 펌웨어를 DK 보드에 작성하고 업 그레이드 후 로그 파일을 생성하는 역할을 한다. 반드시 아래와 같이 이 파일 내용 안에 eddy-bl-2.5.x.x.bin 파일 이름과 Eddy_burning_SerialFlash.tcl 파일 이름이 다운로드 한 파일 이름과 동일하게 작성되어 있는지 확인한다.

```
sam-ba.exe \usb\ARM0 AT91SAM9G20-EK Eddy_burning_SerialFlash.tcl / eddy-bl-2.5.x.x.bin >
logfile.log
notepad logfile.log
```

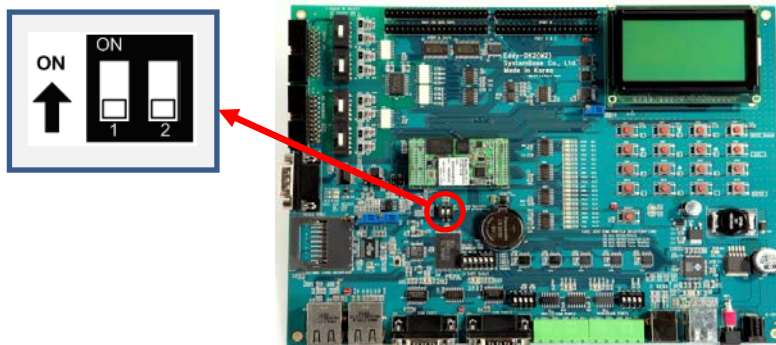
- Eddy_burning_SerialFlash.tcl: 펌웨어 각각의 파일을 보드에 쓰는 역할을 한다. 이 TCL 파일도 마찬가지로 아래와 같이 eddy-bs-2.5.x.x.bin, eddy-os-2.5.x.x.bin, 그리고 eddy-fs-2.5.x.x.bin 파일 이름이 다운로드 한 파일 이름과 동일한지 확인하여야 한다.

```
...
#####
# Main script: Load the linux demo in SerialFlash,
#           Update the environment variables
#####
array set df_mapping {
    bootstrapFileName "eddy-bs-2.5.X.X.bin"
    kernelFileName    "eddy-os-2.5.X.X.bin"
    filesystemFileName "eddy-fs-2.5.X.X.bin"
}
```

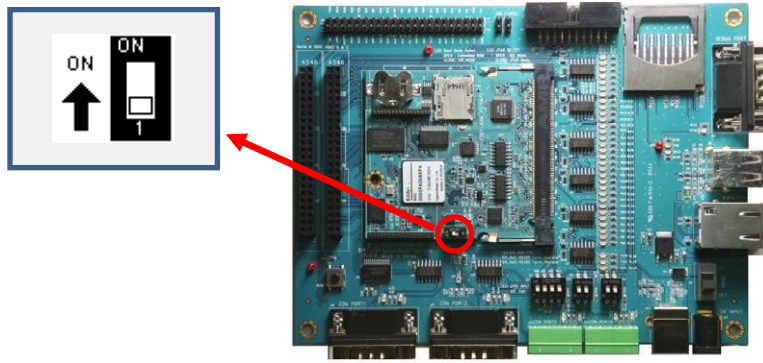
9.2.2 DK 보드 드라이버 설치

PC 에서 Eddy DK 또는 Eddy-S4M-DK 을 USB 디바이스로 인식되도록 하기 위해 다음과 같은 순서로 드라이버를 설치한다

- DK 보드 전원을 끈다.
- USB 케이블로 DK 보드와 PC 를 연결한다.
- 아래 그림에 표시된 DIP 스위치를 Off시킴으로써 USB 대기모드로 설정한다.



[Eddy-CPU DK인 경우(S6 DIP스위치)]



[Eddy-S4M DK인 경우(S1 DIP스위치)]

- DK보드의 전원을 연결한다..
- 정상적으로 PC 에서 DK보드를 인식하면, 새로운 드라이버 설치 준비작업을 위한 다이얼로그 박스가 생성된다.
“예, 장치를 연결할 때마다 연결” 을 선택하고 다음을 선택한다.
- 소프트웨어 자동으로 설치(권장) 을 선택한 후 “다음(N)” 을 선택한다.
- 드라이버가 설치된 폴더를 선택하여 atm6124.sys ATMEL AT91xxxx Test Board 드라이버가 선택한 후 계속(C) 을 선택한다.
- “마침” 을 선택하여 드라이버 설치를 종료한다.
- DK보드의 DIP 스위치를 On 시킨다.

9.2.3 USB 시스템 복구 실행

- DK 보드 전원을 끈다.
- USB 케이블로 DK 보드와 PC 를 연결한다.
- DK 보드의 DIP 스위치를 Off 시킴으로써 USB 대기모드로 설정한다



[Eddy-CPU DK:S6] [Eddy-S4M DK:S1]

- DK 보드의 전원을 켜다.
- 약 5초 후 다시 DK 보드의 DIP 스위치를 ON 시킴으로써 플래시에 데이터가 작성될 수 있도록 설정한다.



[Eddy-CPU DK:S6] [Eddy-S4M DK:S1]

- USB 시스템 복구가 설치된 폴더에서 Eddy_burning_SerialFlash.bat 배치 파일을 실행함으로써 USB 복구를 시작한다. 이 배치 파일이 실행된 후 약간의 시간이 지나면 결과를 확인할 수 있다.
- 업그레이드가 끝난 후에 logfile.log 파일을 통하여 USB 시스템 복구가 성공적으로 되었는지 확인할 수 있다. 아래와 같이 성공적인 로그 메시지가 아닌 경우 “9.2.4 USB 시스템 복구 문제점 해결”을 참고하여 문제를 해결한다.

```
...

u-boot file: eddy-bl-2.x.x.x.bin

...

-I- ==== Load the bootstrap: xxxxfldsh_at91sam9xxxek in the first secto
* ----
```

- 정상적인 로그 메시지를 확인한 경우 DK보드의 전원을 리셋 하여 USB 복구한 프로그램으로 부팅되는 것을 확인한다.

9.2.4 USB 시스템 복구 문제점 해결

- USB 복구 파일 이름이 잘못 되었을 경우 아래와 같은 로그 에러 메시지가 출력된다. 이 경우에는 Eddy_burning_xxxxFlash.bat 파일 혹은 Eddy_burning_xxxxFlash.tcl 파일에서의 펌웨어 이름과 다운로드 한 펌웨어 이름이 동일한지 확인한 후 다를 경우 수정하여 다시 설치한다.

```
...

script file : Eddy_burning_xxxxFlash.tcl

u-boot file: eddy-bl-2.x.x.x.bin

-E- Script File Eddy_burning_xxxxFlash.tcl returned error : could not read "eddy-bl-2.x.x.x.bin": no such file or directory - could not read "eddy-bl-2.x.x.x.bin": no such file or directory

while executing
```

- 아래와 같은 로그 에러 메시지가 출력되면 USB 케이블이 PC와 보드가 정상으로 연결되어 있는지 확인한 후 다시 설치한다.

```
-I- Waiting ...

-E- Connection \usb\ARM0 not found
```

- 아래와 같은 로그 에러 메시지가 출력되면 Eddy DK 보드의 S6 DIP 스위치에 있는 오른쪽 스위치가 Down 되어 플래시에 쓸 수 없도록 되어 있는지 확인하고 스위치를 Up 시킨 후에 다시 설치한다.

...

-I- Loading applet isp-dataflash-at91sam9g20.bin at address 0x20000000

-E- Script File Eddy_burning_DataFlash.tcl returned error : Error Initializing
xxxxFlash Applet (Can't detect known device) - Error Initializing xxxxFlash
Applet (Can't detect known device)

while executing


"error "Error Initializing xxxxFlash Applet (\$dummy_err)""

9.3 제품 사양

Eddy 제품 군에 대한 구성은 다음과 같다.

9.3.1 Eddy-CPU v2.1 / v2.5 / v2.5B Specifications

	구분	규격		
		Eddy-CPU v2.1	Eddy-CPU v2.5	Eddy-CPU v2.5B
Hardware	CPU	AT926EJ-S (210 MHz)	AT91SAM9G20 (400 MHz)	
	Memory	8MB Data Flash,		
		32 MB SDRAM		64MB SDRAM
	External I/F	19 Bit / 16 Bit Data Bus		
	Ethernet I/F	10/100 Base-T Auto MDI/MDIX		
	UARTs	4 Port, Support up to 921.6 Kbps (1 : Full Signal, 2,3,4, : RxD, TxD, RTS, CTS only)		
	USB 2.0 FS	2 Host /1 Device Port, 2.0 FS (12Mbps)		
	ADC	4-Channel 10 Bit ADC		
	TWI(I2C)	Master, Multi-Master and Slave Mode		
	SPI	8- to 16-bit Programmable Data Length Four External Peripheral Chip Selects		
	GPIO	Max. 56 Programmable I/O Pins		
	Power Input	3.3 V (200 mA Max)		
	Dimensions	25 x 48.5 x 6.2 mm		
	Weight	8.3 g		
Network	Protocol	TCP, UDP, Telnet, ICMP, DHCP, TFTP, HTTP, SNMP 1&2, SSH, SSL		
	Ethernet	10/100Mbps MAC / PHY		
	Network Connection	Static IP, DHCP		

	구분	규격		
		Eddy-CPU v2.1	Eddy-CPU v2.5	Eddy-CPU v2.5B
Software	O/S	Linux Kernel 2.6.21		
	Mgt Tools	SNMP, Web, PortView		
	Uploads	TFTP, FTP, Web		
	Dev Tools	LemonIDE & SDK		
Environmental	Operating Temp	-40 ~ 85 °C		
	Storage Temp	-60 ~ 150 °C		
	Humidity	5 ~ 95% Non-Condensing		
Approvals	CE Class A, FCC Class A, RoHS compliant			

9.3.2 Eddy-DK v2.1 Specifications

구분	규격
NAND Flash	256MB, 8bit I/F
SD Card	Push Type, Up to 16 GB
Connector	MMC / SD Card / MC supported
USB Connector	1 x Device 2 x HOST, Dual-Port
LCD Module	128 x 64 Dots Matrix Structure
KEY	4 x 4 Matrix
Battery Holder	3V Lithium Battery, 235 mAh
LED	Power, Ready, 20 Programmable IO, Console & Serial TxD, RxD
I2C Interface	16bit I2C BUS GPIO
SPI Interface	2 Kbit EEPROM
MCI Interface	SD Card, MMC Socket
ADC Interface	Temp / Light Sensor

구분	규격
Digital I/O	8 Port Input, 8 Port Output
Switch	<ul style="list-style-type: none"> - Serial or GPIO Select - RS422/485 Select - DIO : Common VCC or GND Select - Programming
Jumper Switch	Boot Mode Select, JTAG Select
Serial Port	2 x RS232 DB9 Male 2 x RS422/485 Terminal Block (RS422 & RS485 Selected by S/W)
Console Port	DB9 Male
LAN Port	2 x RJ45
ICE Port	Used for Flash Programming
Reset Button	Factory Default & Warm Boot
Input Power	9-48VDC
Dimensions	240 x 180 mm

9.3.3 Eddy-S4M v2.1 / v2.5 Specifications

	구 분	규 격	
		Eddy-S4M v2.1	Eddy S4M v2.5
Hardware	CPU	AT9260B-CJ (210 MHz)	AT91SAM9G20 (400MHz)
	Memory	AT45DB642D, 8MB Data Flash IS42S16160B, 32 MB SDRAM	
	Ethernet MC/PHY	10/100 Base-T MAC KSZ8041NLI PHYceiver Auto MDI/MDIX	
	Serials	Port 0,1 : RS232 (DB9 male) Port 0 : Full Signal Port 1 : TxD, RxD, RTS, CTS only Port 2,3 : COMBO (Terminal Block 5pin) * COMBO : RS422/RS485 is S/W selectable	
	USB 2.0 FS	3 Host /1 Device Port, 2.0 FS (12Mbps) GL850A USB Hub chip 을 사용하여 확장된 포트이다.	
	RTC	Real Time Clock, RTC DS1340U-33+ I2C I/F 로 연결	
	Battery Holder	CR1220(38mAh) 3V Lithium Battery	
	ADC	4-Channel 10 Bit ADC	
	TWI(I2C)	Master, Multi-Master and Slave Mode	
	SPI	8 to 16-bit Programmable Data Length Four External Peripheral Chip Selects	
	MCI	SD Spec V2.0 [SDHC], MMC Spec V4.2 지원 USB to SD Controller 적용, 16GB, 12Mbps/s	
	GPIO	Max. 34 Programmable I/O Pins	
	LED	Ready LED	
Software	Protocol	TCP, UDP, Telnet, ICMP, DHCP, TFTP, HTTP, SNMP1&2, SSH, SSL	
	Network Connection	Static IP, DHCP	

	구 분	규 격	
		Eddy-S4M v2.1	Eddy S4M v2.5
	O/S	Linux Kernel 2.6.21	
	Mgt Tools	SNMP, Web, PortView	
	Uploads	TFTP, FTP, Web	
	Dev Tools	LemonIDE & SDK	
Physical characteristics	Power Input	3.3 V (200mA Max)	
	Dimensions	59.75 x 61.80 x 4 mm	
	Weight	15 g	
Environment	Operating Temp	-40 ~ 85° C	
	Storage Temp	-66 ~ 150° C	
	Humidity	5 ~ 95% Non-Condensing	
CE Class A, FCC Class A, RoHS compliant			

9.3.4 Eddy-S4M-DK v2.1 Specifications

구 분	규 격
Serial Port	2 x RS232 DB9 Male 2 x RS422/485 5pin Terminal Block (S/W Selectable & with Auto toggle)
SD Card Connector	Push Type, Up to 16 GB MMC / SD Card / MC supported
MCI Interface	SD Card, MMC Socket
ADC Interface	Light Sensor
USB Connector	1 x Device, 2 x HOST, Dual-Port
LAN Port	RJ45 with transformer
Console Port	DB9 Male
Switch	Power ON/Off 스위치 Serial RS422/485 Termination resistor 설정 스위치 GPIO 입력 시험 스위치(Off : Low, ON : High)
LED	RDY, Power, 34 Programmable IO, Console & Serial TxD, RxD LED
JTAG Port	Used for downloading code and single-stepping through programs
Reset Button	Factory Default & Warm Boot (5 초이상 누르고 있으면, Factory default 로 동작)
JIG 연결 소켓	2 2x23pin socket, JIG 보드를 연결하고 이상유무를 시험하기 위한 커넥터
Expansion Header	2x22pin Header, Eddy-S4M 의 GPIO 를 시험하기 위한 커넥터
Input Power	5 VDC
Dimensions	160 x 120 mm

9.3.5 Eddy-S4M-JIG v2.1 Specifications

구분	규격
USB Connector	USB HOST

구분	규격
LAN Port	RJ45
Reset Button	Factory Default & Warm Boot
Expansion Header	S4M 0이 제공하는 모든 기능을 외부장치와 연결 가능하도록 한다.
Input Power	5 VDC
Dimensions	70 x 105 mm

9.3.6 Eddy-WiFi v3.0 Specifications

Classification	Specification
Standard	802.11b, 802.11g, 802.11n
Modulation	802.11b:CCK, DQPSK, DBPSK 802.11g:64 QAM, 16 QAM, QPSK, BPSK 802.11n:BPSK, QPSK, 16-QAM, 64-QAM
Frequency Band	ISM band 2.4GHz ~ 2.4884GHz
Output Power	802.11b:16 dBm (11Mbps) 802.11g:14 dBm (54Mbps) 802.11n:14 dBm (20MHz BW,MCS7) 13 dBm (40MHz BW,MCS7)
RX sensitivity	802.11b:-84dBm@11MHz 802.11g:-73dbm@54MHz 802.11n:-71dBm(MCS 7_HT20) -68dBm(MCS 15_HT20) -68dBm(MCS 7_HT40) -65dBm(MCS 15_HT40)
Security	WPA, WPA-PSK, WPA2, WPA2-PSK , WEP 64bit & 128bit , IEEE 802.11x, IEEE 802.11i
Working distance	60 - 120m, depending on surrounding Environment
Data Rate	802.11b: 11, 5.5, 2, 1 802.11g: 54, 48, 36, 24, 18, 12, 9, 6 802.11n: 20 MHz BW: 130, 1117, 104, 78, 65, 58.5, 52, 39, 26, 19.5,


	13, 6,5 40 MHz BW: 270, 243, 216, 162, 150, 135, 121,5, 108, 81, 54, 40,5, 27, 13,5 (unit: Mbps)
Antenna	ANT 2,4Ghz 2DB, 1 x U,FL
Dimension	28,2 X 45,4 X 9,6 mm
Operating Temp	-10 ~ 70° C
Operating Voltages	3,3V ± 5% I/O supply voltage
Weight	10g
Approvals	KC, RoHS Compliant

9.3.7 Eddy-BT v2.1 Specifications

구 분	규 격
Interface	Bluetooth v2,0+ EDR Class 1
Profile	SPP (Serial Port Profile)
Max, TX Power	+18dBm
RX sensitivity	-88dBm
Power	Supply voltage: 3.3V DC Supply current::10mA – 60mA
Operating Temp	Operating temperature: -30 ~ 80 °C
Storage Temp	Storage temperature: -40 ~ 85 °C
Humidity	Humidity : 90% (Non-condensing)
Working distance	Stub Antenna (+1dBi) - Stub Antenna (+1dBi) 100 meters Stub Antenna (+1dBi) - Dipole Antenna (+3dBi) 150 meters Dipole Antenna (+3dBi) - Dipole Antenna (+3dBi) 200 meters Dipole Antenna (+3dBi) - Dipole Antenna (+5dBi) 300 meters Dipole Antenna (+3dBi) - Patch Antenna (+9dBi) 500 meters Dipole Antenna (+5dBi) - Dipole Antenna (+5dBi) 400 meters Dipole Antenna (+5dBi) - Patch Antenna (+9dBi) 600 meters Patch Antenna (+9dBi) - Patch Antenna (+9dBi) 1,000 meters
Approvals	CE Class A, FCC Class A, RoHS Compliant

9.3.8 Eddy-CPU/mp v2.5 / v2.5 32bit Specifications

	구분	규격
Hardware	CPU	AT91SAM9G20 (400 MHz)
	Memory	8MB Data Flash, 32 MB SDRAM, 64MB SDRAM
	External I/F	16 Bit / 32 Bit Data Bus
	Ethernet I/F	10/100 Base-T Auto MDI/MDIX
	UARTs	4 Port, Support up to 921.6 Kbps (1 : Full Signal, 2,3,4, : RxD, TxD, RTS, CTS only)
	USB 2.0 FS	2 Host /1 Device Port, 2.0 FS (12Mbps)
	ADC	4-Channel 10 Bit ADC
	TWI(I2C)	Master, Multi-Master and Slave Mode
	SPI	8- to 16-bit Programmable Data Length Four External Peripheral Chip Selects
	GPIO	Max. 56 Programmable I/O Pins
	Power Input	3.3 V (200 mA Max)
	Dimensions	59.75 x 44.6 X 1.0 mm
	Weight	8.3 g
Network	Protocol	TCP, UDP, Telnet, ICMP, DHCP, TFTP, HTTP, SNMP 1&2, SSH, SSL
	Ethernet	10/100Mbps MAC / PHY
	Network Connection	Static IP, DHCP
Software	O/S	Linux Kernel 2.6.21
	Mgt Tools	SNMP, Web, PortView
	Uploads	TFTP, FTP, Web
	Dev Tools	LemonIDE & SDK
Environmental	Operating Temp	-40 ~ 85 ℃
	Storage Temp	-60 ~ 150 ℃
	Humidity	5 ~ 95% Non-Condensing

	구분	규격
Approvals	CE Class A, FCC Class A, RoHS compliant	

9.4 주문 정보

Eddy 제품군에 대한 주문정보는 다음과 같다.

제품명	버전	설명
Eddy-CPU	2.1	Embedded CPU Module 32MB
Eddy-CPU	2.5	Embedded CPU Module 32MB
Eddy-CPU	2.5B	Embedded CPU Module 64MB
Eddy-DK	2.1	Eddy V2.1 Development Kit
Eddy-S4M	2.1	Embedded CPU Module (Mini PCI Type)
Eddy-S4M	2.5	Embedded CPU Module (Mini PCI Type)
Eddy-S4M-DK	2.1	Eddy-S4M v2.1 Development Kit
Eddy-S4M-JIG	2.1	Eddy-S4M v2.1 JIG Board
Eddy-WiFi	3.0	802.11 b/g/n WiFi Module
Eddy-BT	2.1	Bluetooth 2.0 Module
Eddy-CPU/mp	2.5	Embedded CPU Module (32MB SDRAM)
Eddy-CPU/mp 32bit	2.5	Embedded CPU Module (32MB SDRAM)
Eddy-CPU/mp 32bit	2.5	Embedded CPU Module (64MB SDRAM)
Eddy-CPU/mp-JIG	2.5	Eddy-CPU/mp v2.5 JIG Board