▣ LOW_POWER CONCEPT

2020.11.29 강경수

1. 개요

-. 무선 키보드에서 일반적으로 건전지를 사용하기에 반드시 전력소모에 대해서 고려하여야 할 필요가 있다.

-. 이때 가장 먼저 고려 되어야 하는 것은 MCU 자체의 전류소모를 줄이는 일이다.
또한 주변 소자들 소모 전력도 고려하여 최적화된 전력소모 시나리오를 찾아야 한다.

2. 주변 회로 전류 계산

Components	MCU	LDO	LED	1602 LCD	RF IC	TOUCH IC	TOTAL
Part Name	ATMEGA328p	1uA			NRF24L01	MTCH6102	285uW
Condition	3.3V	3.6V	3.3V	5V	3.3V	3.3V	
Operating Ratio	하루 1만회 TX	100%			하루 1만회 TX	5%	
Low Power Mode	1uA	-	-		900nA	500nA	
Active Mode	7mA/15mA	1uA	10mA	150mA	11.3mA	12uA	
POWER (mW)	178.083uW	3.3uW			63.53uW	39.6uW	

※ 3.6V COINBATTERY 2개 병렬 기준 (3.6V * 1000mAh * 2 = 7,200mW/h)
7,200 / 0.285 / 24(HOUR) / 29.5(DAY) = 35.68 즉 약 3년 사용 가능

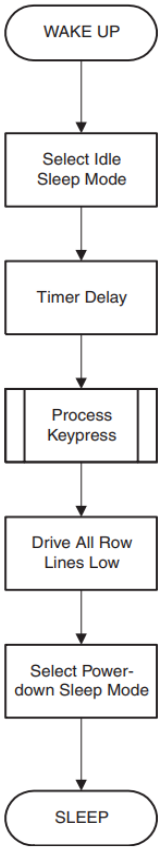
3. MCU KEY SCAN 동작 시나리오

기본은 Sleep idle, Mode에서 동작하며
이때 I/O PORT제어 및 SPI통신 모두 가능하다

또한 키채터링 현상을 펌웨어 적으로 해결하기 위하여
키 입력시 Timer Delay를 일정시간 준다

키 입력에대한 처리 완료후

Sleep Mode에 다시 들어가기전에 반드시 모든
Row핀들을 Low 상태로 만들어준다!
(그래야 모든 키입력에 대하여 Wakeup 할 수 있음)



4. TOUCH SCAN 동작 시나리오

미정

5. NRF24L01 TIMING 계산

SEQUENCE	Execution Time
POWER DOWN → STANDBY 1 MODE	150us
STANDBY 1 → TX MODE	130us
TRANSMIT DATA → CE LOW DELAY	14us
Total Time	65.7ms

OPERATION MODE	TIME
STANDBY1 MODE	0.5% (15mA)
Idle sleep	99.5% (7mA)
Power-down	그 외 모든 시간 1uA

6. MCU CODE TIMING 계산

SEQUENCE	Execution Time
PIN CHANGE WAKE UP -> INIT TIMER -> ENTER IDLE SLEEP	40us
IDLE SLEEP -> TIMER OVER FLOW	65.4ms
TIMER WAKE UP->KEYBOARD SCANNING->ENTER POWER DOWN	351us
Total Time	65.7ms

※ System Clock : 8Mhz

OPERATION MODE	TIME
Normal	0.5% (15mA)
Idle sleep	99.5% (7mA)
Power-down	그 외 모든 시간 1uA

즉 하루 1만타 입력기준 :

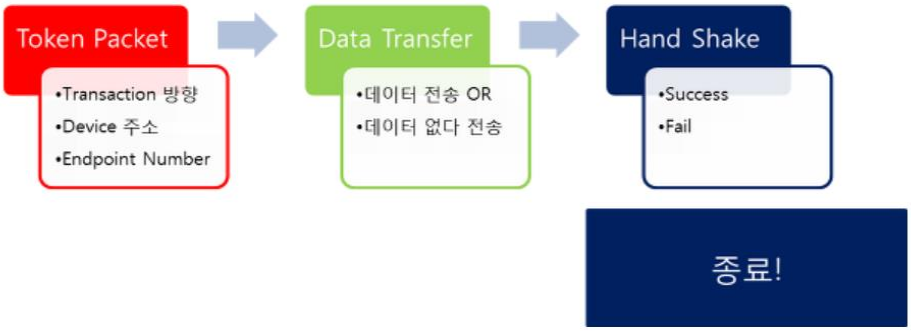
(((15mA X 3.3 X 0.005 + 7mA X 3.3 X 0.995) X 0.0657) X 0.11574)+(1uA X 3.3 X 0.9343)

=178.083uW

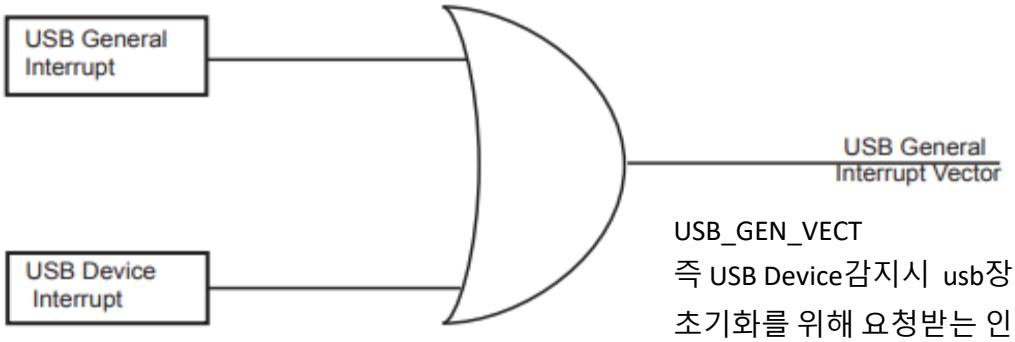
▣ USB 2.0 HID CLASS

2020.11.27 강경수

1. USB 통신의 기초



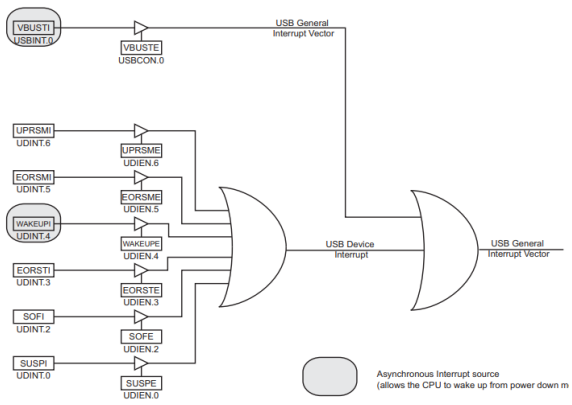
2. ATmega32u4 USB와 관련된 INTERRUPT 2가지



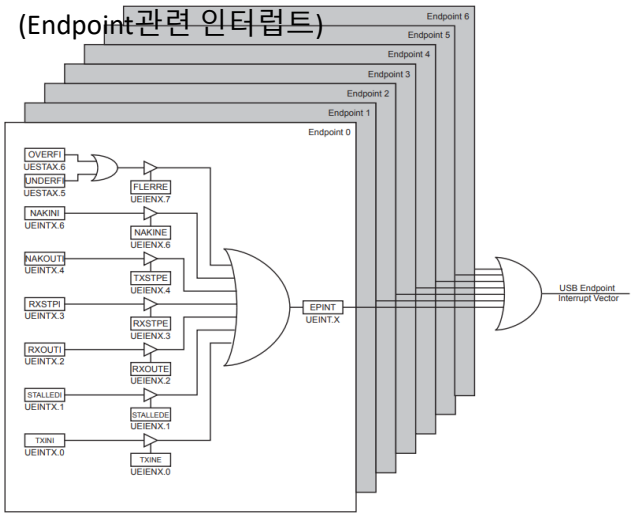
USB_GEN_VECT
즉 USB Device감지시 usb장치 초기화를 위해 요청받는 인터럽트

Endpoint/Pipe ,즉 USB장치의 순수 DATA전달 목적을 위한 인터럽트

USB DEVICE그 자체에 관한 세팅을 위한 인터럽트



DEVICE가 전달한 DATA에 관한 INTERRUPT (Endpoint관련 인터럽트)



3. PLL이란 무엇인가?

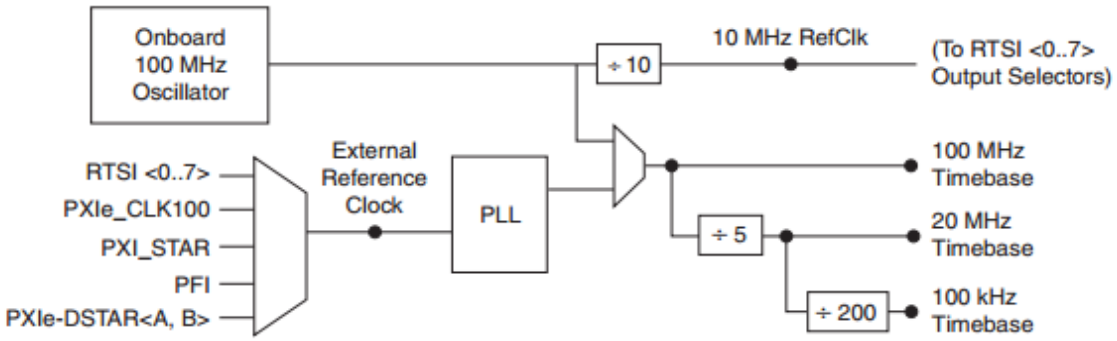
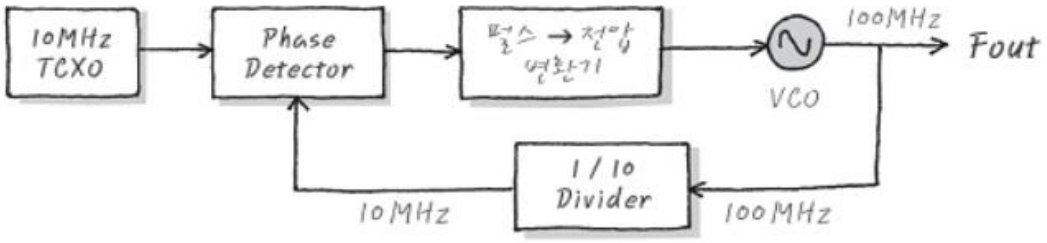


그림 1. X 시리즈 타이밍 소스



1. 주파수원이 흔들리지 않도록 고정해준다.

예) : 836.5MHz의 주파수원이 주변환경에 따라 흔들려서 832MHz나 840.2MHz 처럼 이상한 주파수로 가 버리면 정상적인 RF신호처리가 될리가 만무하쥐~

2. 주파수원을 정확하게 가변한다.

예) 하나의 시스템에서 836.5MHz의 주파수원을 1.2MHz 간격으로 다른 주파수로 바꾸고 싶은데.. 837.7 MHz, 838.9 MHz 뭐 이런 식으로 하고 싶을 때...

USB통신에서 일반 ON Board 상의 16MHz에 비하여 높은 주파수 32~96Mhz를 필요로 한다. uP에 이런 주파수를 만들어 줄 수 있는 PLL회로가 내장돼 있다.
Control Register setting을 통해 원하는 USB통신 주파수를 사용 할 수 있다.

4. USB Endpoint란 무엇인가?

-.Endpoint란 USB Device의 Buffer이다. 호스트 운영체제와 독립적으로 하드웨어 Level에서 지원하는 Buffer이다. Host는 이 Endpoint를 통해서 데이터를 주고 받을 수 있는것이다.

-.Atmega32u4 Endpoint는 0~6번까지 지원한다.

-.모든 Endpoint0은 제어 용도로 사용 된다.

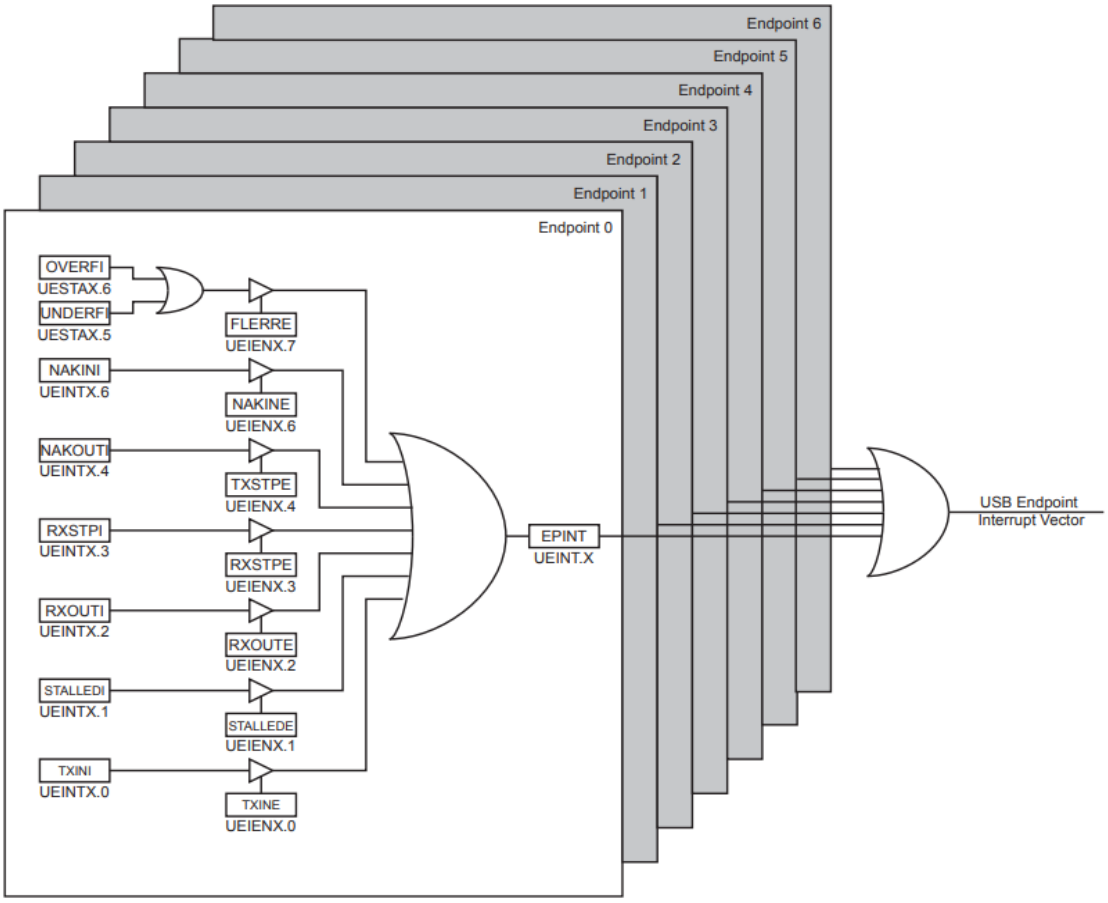
-.모든 Endpoint는 양방향이며 즉 Host는 데이터를 전송할 수 있고 Endpoint에 쌓여 있는 Data를 한번에 수신 할 수 있다.

-.Endpoint0의 목적은 Host가 Device의 정보를 얻고, 구성하고, 제어작업을 수행하기 위함이다. (일반적인 Data 전송 목적이 아님)

-. 즉 데이터 전송을 위한 메모리 공간을 Endpoint, 이때 Host와 EP사이의 전송모델을 Pipe라 한다.

EP = PIPE가 아니라 EP는 메모리 공간이며 이 통로의 모델링 구조를 **PIPE**라 한다

(자세한 내용은 아래 전송형태 참조)



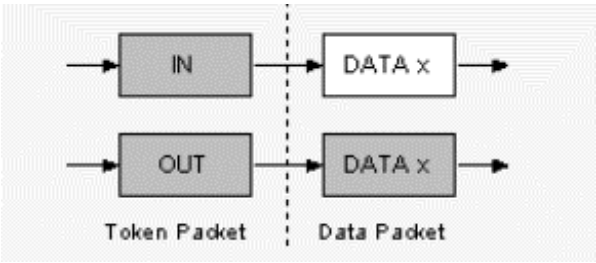
5. USB 전송형태(Transfer Type)

①USB통신의 기본: IN AND OUT (IN : Host가 Device로에게 DATA요청
OUT : Host가 Device에게 DATA 보냄)

※ 하얀색 : Device가 보내는 Packet
※ 검정색 : Host가 보내는 Packet

② Isochronous Transfer(등시성 전송)

-.데이터 전송시간을 보증하기 위해서 사용 데이터 오률 있어도 재전송 안함.
Host또한 등시선 전송을 위해 시간 스케쥴링함
실시간이 중요한 디바이스에 사용(ex : 이어폰)

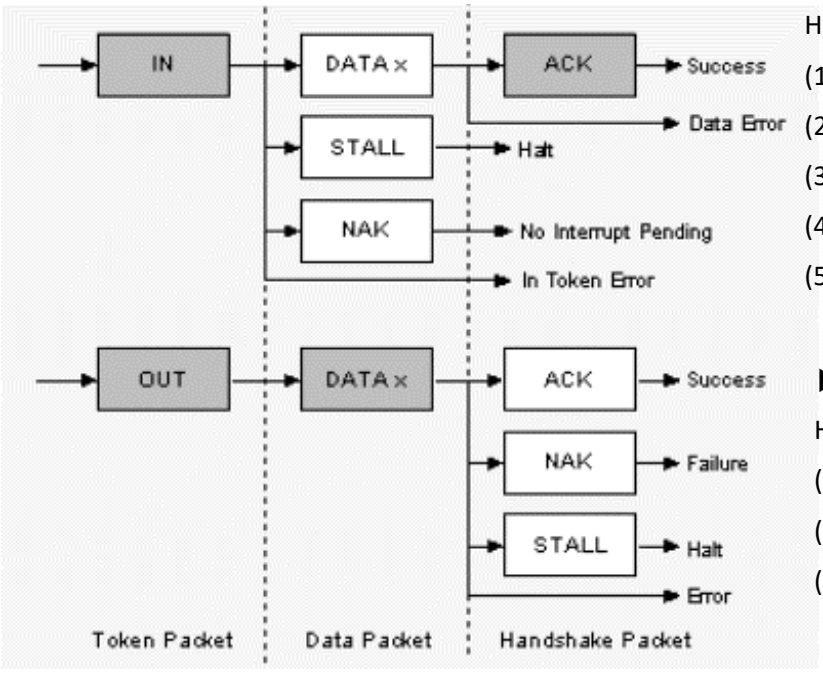


③ Interrupt Transfer(인터럽트 전송)

-.작은 크기의 데이터를 전달할 때 사용하는 타입이다. (키보드,마우스 등)

-. 인터럽트 요청은 디바이스에 의해서 큐에 담기고, 호스트가 USB디바이스에게 "너 보내줄 데이터 있니?"하면 큐에담긴 인터럽트 요청들이 디바이스에서 호스트로 보내진다.

-. 키보드로 예를 들면 키보드에 입력한 데이터들이 디바이스의 내부 메모리 큐에 저장되고 호스트가 Polling하면 호스트에서 우리가 입력한 문자들을 받아들임
(Host Polling 주기는 Endpoint Descripotr에 서 결정 함)



▶IN TOKEN PACKET
HOST가 DATA를 받는경우.
(1) HOST IN PACKE전송
(2) DEVICE 큐에 저장된 DATA 전송
(3) 잘 수령시 ACK
(4) 큐가 비어있으면 NAK
(5) 오류 발생시 STALL →
다시 IN PACKET 전송

▶OUT TOKEN PACKET
HOST가 DATA를 보내는 경우.
(1) HOST OUT PACKET 발행
(2) EndPointBuffer에 DATA저장
(3) 이전 Packet처리 못해서
Buffer 비어잇는 상태 아니면
NAK 오류상황일때는 STALL

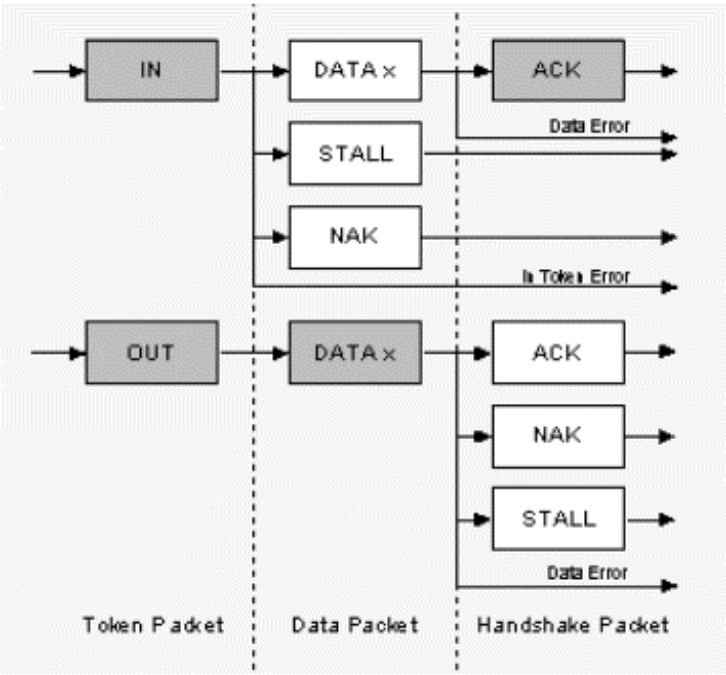
④ Bulk Transfer(벌크 전송)

-.대량의 데이터를 신뢰성 있게 전달할 때 사용하는 타입이다. 데이터 패킷 전송과정에서 오류가 발생하면 재전송을 요구하며, 등시성 전송과는 다르게 전송 속도는 보증하지 않는다.
대표적으로 USB 저장 장치(Mass Storage) 가 있다.

-. 전송속도보다는 전송 데이터의 질을 요구하는 Device에 사용된다.

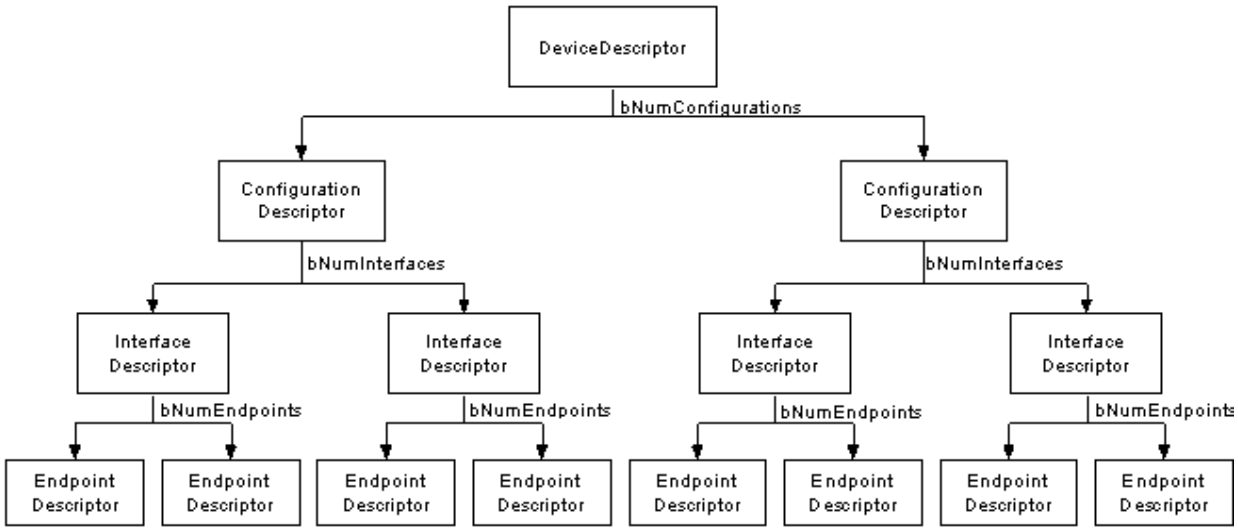
-. Bulk Transfer와 Interrupt Transfer는 기본적으로 같은 트랜잭션 형태를 보이지만 , Interrupt Transfer는 Polling 주기를 설정 할 수 있다는점에서 다르다.

-. 마우스의 경우 0.2초뒤에 클릭이 된 것처럼 처리가 되면 불편하지만, 저장장치에서는 상관이 없다, 더불어 Bulk Transfer에서는 1번의 트랜젝션으로 전달이 다 되지 않을 수 있기에 분할해서 트랜잭션을 반복한다.



▣ USB 장치 보고서에 대한 이해

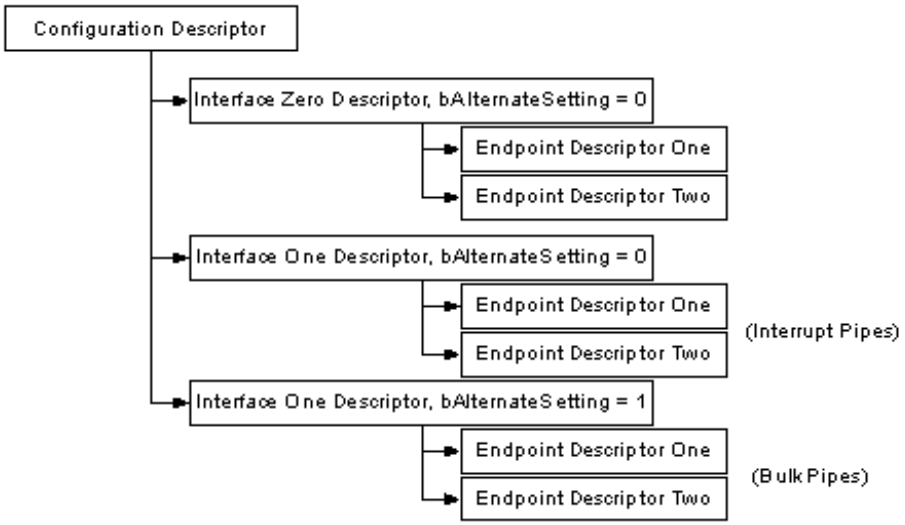
2020.12.02 강경수



-. Device Descriptor : USB 장치에 대한 일반적인 정보들이 담기게 된다. 즉 벤더사, 제조사, USB버전, Configuration Descriptor 개수 등등

USB Specification 2.0문서를 보면 아래와 같이 Description과 함께 설명 돼 있다.
또한 MS, Keil 등 관련 문서를 잘 풀어서 설명해놓은 곳이 많으니 참고하면 좋다.

Offset	Field	Size	Value	Description
6	*bDeviceProtocol*	1	Protocol	Protocol code (assigned by the USB-IF). These codes are qualified by the value of the *bDeviceClass* and the *bDeviceSubClass* fields. If a device supports class-specific protocols on a device basis as opposed to an interface basis, this code identifies the protocols that the device uses as defined by the specification of the device class. If this field is reset to zero, the device does not use class-specific protocols on a device basis. However, it may use class-specific protocols on an interface basis. If this field is set to FFH, the device uses a vendor-specific protocol on a device basis.
7	*bMaxPacketSize0*	1	Number	Maximum packet size for endpoint zero (only 8, 16, 32, or 64 are valid)
8	*idVendor*	2	ID	Vendor ID (assigned by the USB-IF)
10	*idProduct*	2	ID	Product ID (assigned by the manufacturer)
12	*bcdDevice*	2	BCD	Device release number in binary-coded decimal
14	*iManufacturer*	1	Index	Index of string descriptor describing manufacturer
15	*iProduct*	1	Index	Index of string descriptor describing product
16	*iSerialNumber*	1	Index	Index of string descriptor describing the device's serial number
17	*bNumConfigurations*	1	Number	Number of possible configurations



-. Configuration Descriptor : Device가 HOST 전원을 사용할 경우 POWER량, Interface 개수 를 Host에게 알려준다.
대부분의 주요 USB장치들은 Configuration 를 오직 한가지만 갖지만 경우에 따라서 두가지 이상 가질 수 있다.

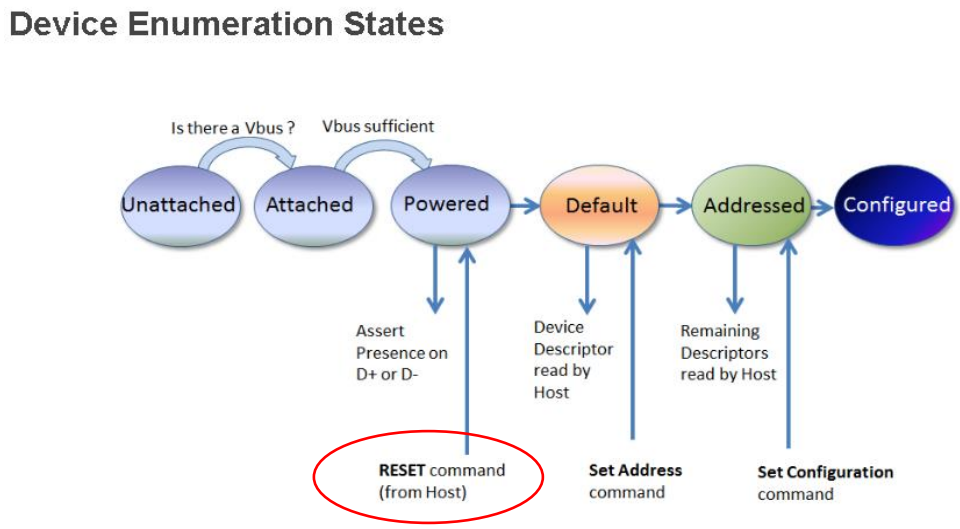
-. Interface Descriptor : 인터넷상에서 구한 코드 분석하며 마저 작성 예정

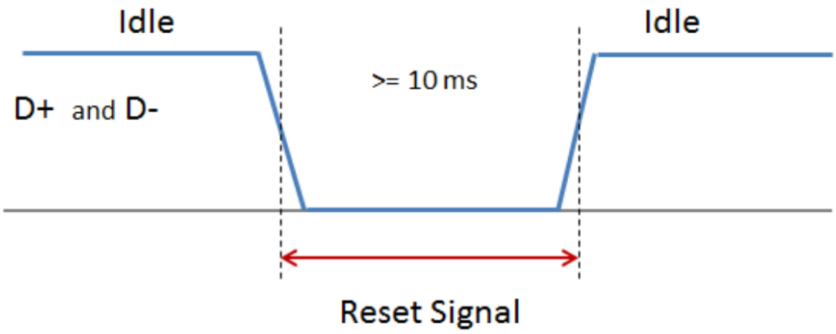
▣ USB Enumeration 이란?

2020-12-03 강경수

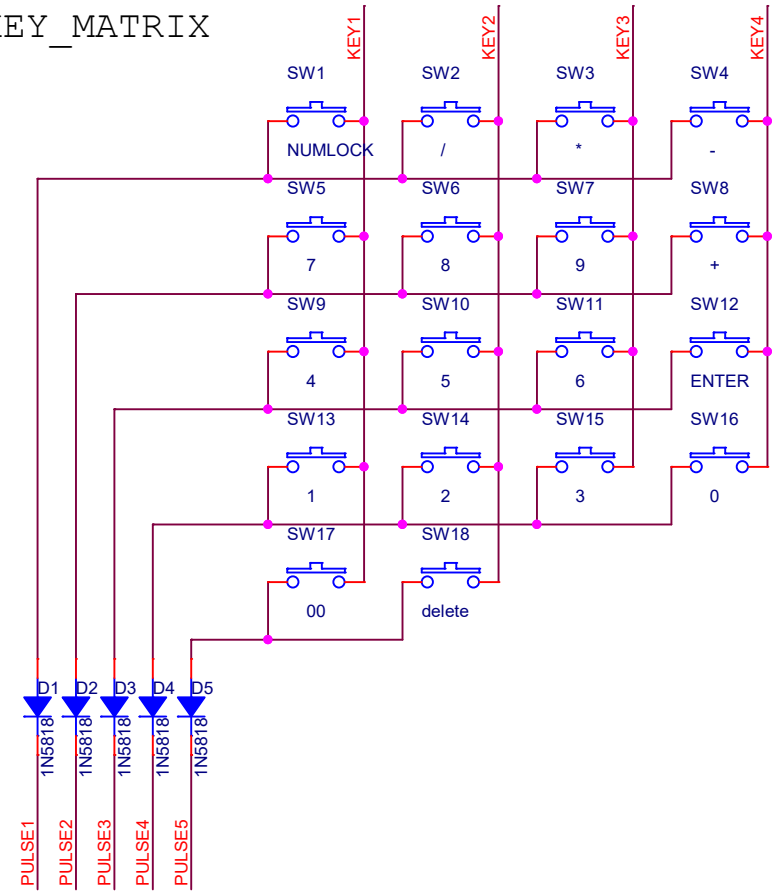
1. Enumeration이란 무엇인가?
-. Host에서 USB를 인식하는 초기 과정

2. PC(Host) 관점
-. Enumeration과정에서 Host는 Device에게 reset을 요청하고 또 Device descriptor를 요청하고 또 Configuration Descriptor를 2번 요청한다.
(2번 요청하는 이유 : 처음에는 공간할당을 위해 크기 데이터만 획득)
-. 이때 HOST에 의해 이뤄지는 RESET은 H/W RESET, POWER ON RESET 개념과는 조금 다르다
-. 아래 그림에서 보듯이 D+ 와 D-RK 10ms이하로 LOW LEVEL로 떨어지면 RESET을 요청하는
-. HOST에서 연결 할 수 있는 Device개수는 127개
-. 왜? 128개도아닌 127개일까 이유는 0번째는 맨 처음 연결된 USB Device가 0번의 Address를 갖기 때문이다.
-. 처음에는 무조건 0번지로 잡고 있다가 Device Descritor에서 0 endpoint사이즈를 알아낸 이후 다른 Address로 할당한다.





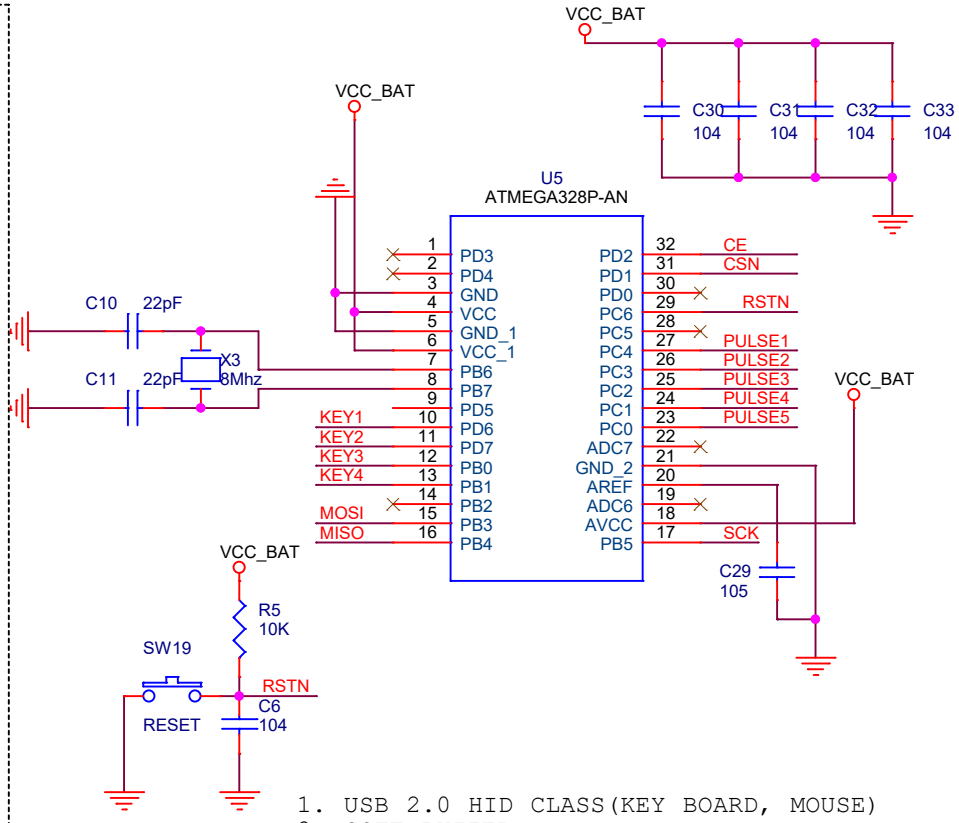
KEY_MATRIX



TOUCH IC

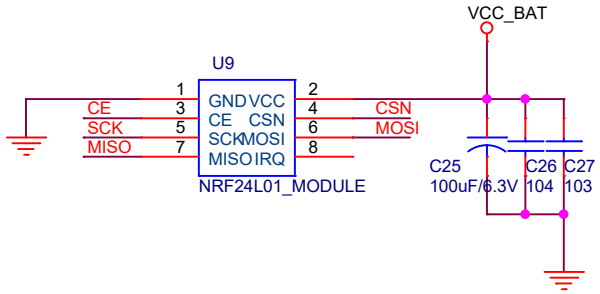
FOR MOUSE

TO BE DETERMINED

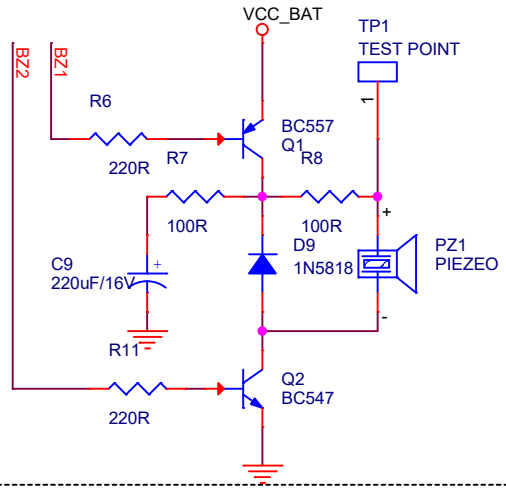


1. USB 2.0 HID CLASS(KEY BOARD, MOUSE)
2. SOFT BUZZER
3. KEY MATRIX(LOW POWER)
4. TOUCH MOUSE(LOW POWER)
5. NRF24L01 2.4Ghz RF COMMUNICAITION

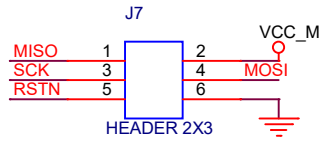
RF MODULE / ISP



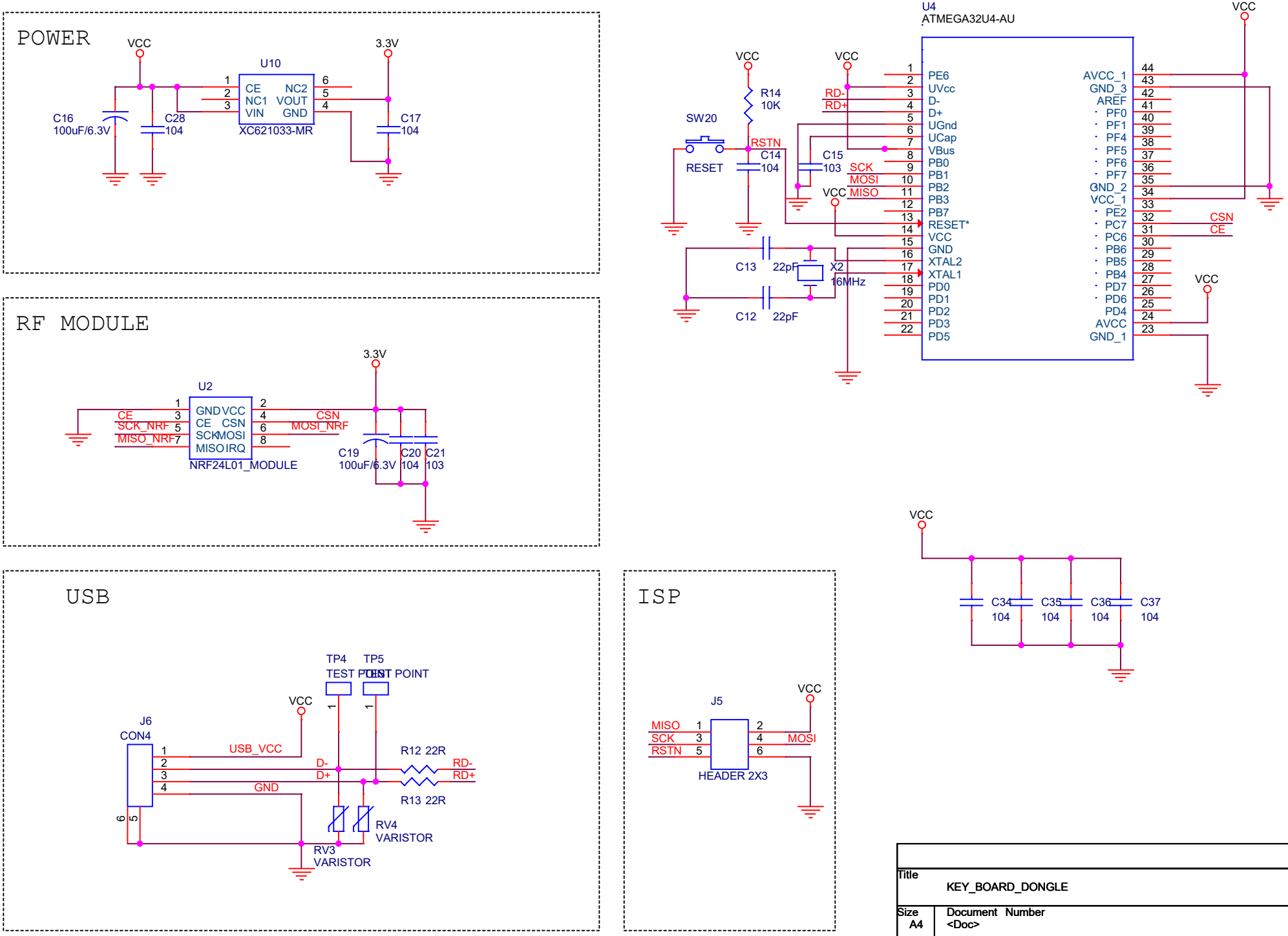
BUZZER



ISP



Title	MY_MACRO_KEYBOARD_VER0.1	
Size A4	Document Number Designed by K.K.S	Rev <RevC
Date: Friday, December 04, 2020	Sheet 1 of 1	


POWER
VCC
U10
3.3V
C16
100uF/6.3V
C28
104
CE
NC1
VIN
NC2
VOUT
GND
XC621033-MR
C17
104
RF MODULE
U2
3.3V
GND
CE
SCK
MISO
VCC
CSN
MOSI
IRQ
CE
SCK_NRF
MISO_NRF
CSN
MOSI_NRF
NRF24L01_MODULE
C19
100uF/6.3V
C20
104
C21
103
USB
TP4
TP5
TEST POINT
VCC
J6
CON4
USB_VCC
D-
D+
GND
R12 22R
R13 22R
RD-
RD+
RV4
VARISTOR
RV3
VARISTOR
ISP
J5
VCC
MISO
SCK
RSTN
MOSI
HEADER 2X3
U4
ATMEGA32U4-AU
VCC
R14
10K
SW20
RESET
RSTN
C14
104
C15
103
SCK
MOSI
MISO
RD-
RD+
PE6
UVcc
D-
D+
UGnd
UCap
VBus
PB0
PB1
PB2
PB3
PB7
RESET*
VCC
GND
XTAL2
XTAL1
PD0
PD1
PD2
PD3
PD5
C13
22pF
X2
16MHz
C12
22pF
AVCC_1
GND_3
AREF
PF0
PF1
PF4
PF5
PF6
PF7
GND_2
VCC_1
PE2
PC7
PC6
PB6
PB5
PB4
PD7
PD6
PD4
AVCC
GND_1
CSN
CE
C34
104
C35
104
C36
104
C37
104
Title
KEY_BOARD_DONGLE
Size
A4
Document Number
<Doc>
Rev
<RevCode>
Date:
Saturday, December 05, 2020
Sheet
1
of
1