\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* motivation \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* I want to work with low cost neoway Chinese gsm modules
* My first task is to test its different AT commands but for that it should be powered up
* Now I will communicate with it by atmega1284p based board which is custom
* So I need usb asp
* Usbasp installation in windows 10 is tricky
* <https://netmaxtech.com/how-to-make/install-usbasp-driver-windows-8-and-windows>
* then i will use avrdudes sw to burn
* <http://blog.zakkemble.co.uk/avrdudess-a-gui-for-avrdude/> installation of it generates error in windows 10 as libusb not found so installation steps in the links should be followed
* first I am going to use arduino ide for the unit tests ‘
* then I will use vMicro in avrstudio 7 to test a compleset class like code
* after that I will merge it with knollery’s mqtt library
* now lets check my custom board with usb asp
* preset saved as commModule atmega1284p read ok
* now lets connect the ftdi usb module which will be used for debug
* checking device manager it was also not installed
* download driver from here <http://www.ftdichip.com/Drivers/D2XX.htm>
* driver install complete

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* mcu setup \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* now we are going to test the comm module custom board neoway gsm with atmega 1284P
* but now I want transform atmega 1284p as an arduino
* sniffing web I got microduino which has atmega 1284p support <https://wiki.microduino.cc/index.php/Microduino-Module_Core%2B>
* it has 2 serial (hardware)
* now lets install this microduino core to our arduino <https://github.com/MCUdude/MightyCore> download zip then extract to hardware folder of sketchbook and install it manually or do it with Arduino IDE by using this tutorial <https://learn.adafruit.com/add-boards-arduino-v164/setup> you have to add this json link <https://mcudude.github.io/MightyCore/package_MCUdude_MightyCore_index.json> then go to boards manager in arduino ide and install mightycore by this the core actually goes to \Users\User\AppData\Local\Arduino15\packages\MightyCore\hardware\avr\1.0.7\cores\MightyCore

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*mcu in a custom board info \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* Now lets check a simple sketch on the hardware
* Before that turn on verbose output and ll compiler error output enable in arduino IDE
* Now our custom board has no ftdi on it so no upload via bootloader so we will use usbasp to burn code to it . this custom boards dev kit board sch files are at hardware folder to get some connection idea
* RGB Led is at 13,14,15 pins
* The atmega1284p as an arduino is also at /hardware folder
* So lets see whether a sketch works or not

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*testing mcu by arduino core installed \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* Blinky example opened
* New sketch created
* Board setup is 8MHz external, bod level 1.8V, pin standard , mcu 1284p , LTO disabled
* Using usbasp burn by arduino ide fails
* So get hex from temp and burn
* Fuse is LowFuse=DC HighFuse=D7 ExtendedFuse =FE for this custom board
* Blinky test worked

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* softserial check \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* Now let’s check debug output on pin 36(PA4) of atmega1284p which is A4/28 in mightyCore
* New sketch is prepared from softwareSerial example of atmega1284
* **MISO pin was used as TX and MOSI as RX So these are the Debug port which we will use**
* It worked perfectly
* Oh pins\_arduino.h at C:\Users\User\AppData\Local\Arduino15\packages\MightyCore\hardware\avr\1.0.7\variants\standard had this

#define digitalPinToPCICR(p) (((p) >= 0 && (p) < NUM\_DIGITAL\_PINS) ? (&PCICR) : ((uint8\_t \*)0))

#define digitalPinToPCICRbit(p) ( (p) <= 7 ? 1 : (p) <= 15 ? 3 : (p) <= 23 ? 2 : 0 )

#define digitalPinToPCMSK(p) ( (p) <= 7 ? &PCMSK1 : (p) <= 15 ? &PCMSK3 : (p) <= 23 ? &PCMSK2 : &PCMSK0 )

#define digitalPinToPCMSKbit(p) ((p) % 8)

Actually no changes were made just given here because one arduino thread told this .h problem but I aint got any prob

* So now we can give command to neoway and it’s reply can be seen on debug

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* neoway Commands check \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* To check the command first I need to know how to power up this thing
* Now from the board design sch there is 2 control pins in the mcu ….these are : REGULATOR enable pin ON/OFF and GSM SWITCH ON OFF
* So now we will write a simple sketch which will power on the GSM module and wait for incoming string from debug port then will send it to neoway and rcv its reply then it will print that reply to the debug port
* Now lets see which pin is REGULATOR ENABLE and which one is GSM SWITCH

**#define REG\_EN\_PIN 3**

**#define GSM\_SW\_PIN 2**

* MIC29302 is the regulator that we are going to control. From MIC29302 you will see that its Enable pin is active high that’s why we have pulled it down to low by a resistor . So no inverted logic for it
* the On/OFF pin of Neoway is active low and a transistor is used to control it so here it is an inverted logic there is a weak pull up resistor for this pin in the gsm module

|  |  |
| --- | --- |
|  | void pinsInit(void)  {  pinMode(REG\_EN\_PIN, OUTPUT);  pinMode(GSM\_SW\_PIN, OUTPUT);  //giving the inactive stages at first  digitalWrite(GSM\_SW\_PIN, HIGH); //active low(~)  digitalWrite(REG\_EN\_PIN, LOW); //active high  delay(500);  }  void powerOn(void)  {  digitalWrite(REG\_EN\_PIN, HIGH);  delay(300);  }  void powerOff(void)  {  digitalWrite(REG\_EN\_PIN,LOW);  } |

* Ow I forgot to tell you gsmModule is connected to mcu via serial1