PROJECT PROPOSAL

This document has the purpose of illustrating the project proposal for the **EMBEDDED SYSTEMS AND DESIGN CLASS.**

As we all know playing a musical instrument like the piano or the keyboard requires the use of both hands, the left hand used to create bass accompaniments and the right hand used to command the leading melody or harmony. The combination of the left and right hand give sounds on two sections or ends of the frequency spectrum which sum up to give a filled sound and complete sound.

For example if the pianist with her right hand is playing a Cmajor chord (C_6 E_6 G_6) , with her left hand she adds some bass notes such as (C_4 and G_4) to give the chord a better and complete feeling.

So what if an individual with a physical impairment of lacking one arm (example the left arm) wants to the play the keyboard ?

As engineers we can offer limited solutions to these individuals where they can obtain both bass sounds and the leading melody or harmony contemporarily, But this has to be under the playing technique of just the left arm creating the bass notes to accompany whatever is played by the right hand.

We approach a solution to these problem with **MIDI KEYBOARDS:**

PROJECT PROPOSAL

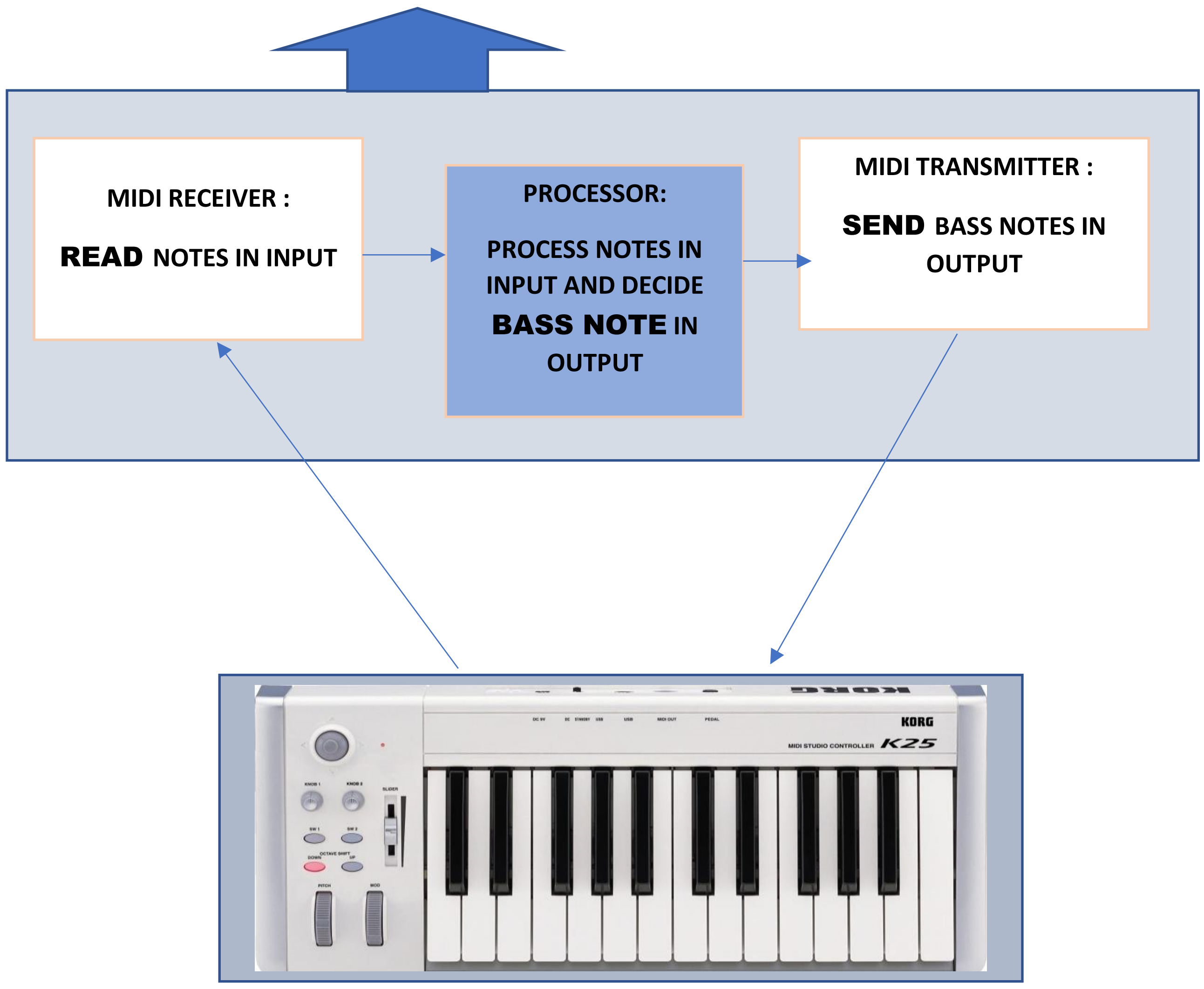
The midi protocol makes this problem simple to define and approach, basically **the problem becomes building a real time object or bridge that reads the notes played by the right and, interprets them and plays bass notes to accompany these notes played by the right hand therefore this object behaving as bridge mimics the left arm.** This entire project proposes a solution for electronic keyboards and not mechanical pianos.

FLOW CHART OF THE PROPOSED SOLUTION

So as discussed earlier the goal is to build a bridge like object that decides the bass note to play based on the input notes it receives, the reception of the input notes, the computation of the bass notes and the transmission of the computed bass notes have to be done in little time in the ratio of microseconds so that the best experience is guaranteed.

The bridge-like object to be built will be called a **MED-BASSER**
And it contains 3 parts for the 3 fundamental operations to be performed.
THE **three** PARTS ARE : **1) MIDI RECEIVER 2) THE PROCESSOR and 3) THE MIDI TRANSMITTER**

OBJECT TO BE BUILT IN THIS PROJECT


MIDI RECEIVER :
READ NOTES IN INPUT
PROCESSOR:
PROCESS NOTES IN INPUT AND DECIDE BASS NOTE IN OUTPUT
MIDI TRANSMITTER :
SEND BASS NOTES IN OUTPUT
KORG
MIDI STUDIO CONTROLLER K25

PROJECT PROPOSAL

Components for the MED-BASSER are

1 X 6N138 Photo-Darlington Transistor OPTOISOLATOR

2 X MIDI DIN CONNECTOR

4x 220 OHM RESISTOR AND 4.7 KOHM RESISTOR

3 x LED

SOME JUMPER WIRES

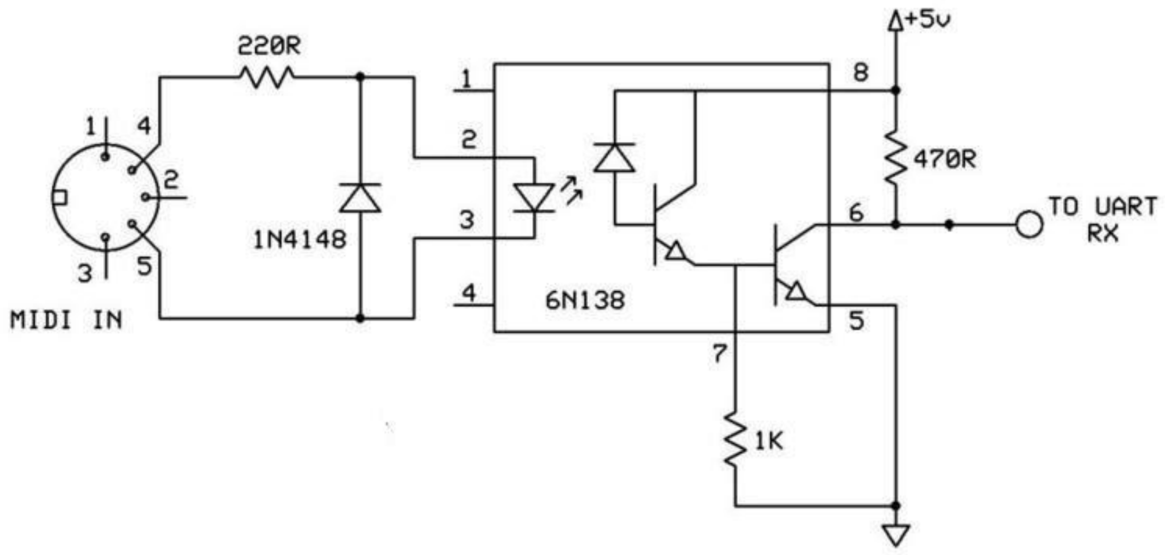
A BREADBOAD

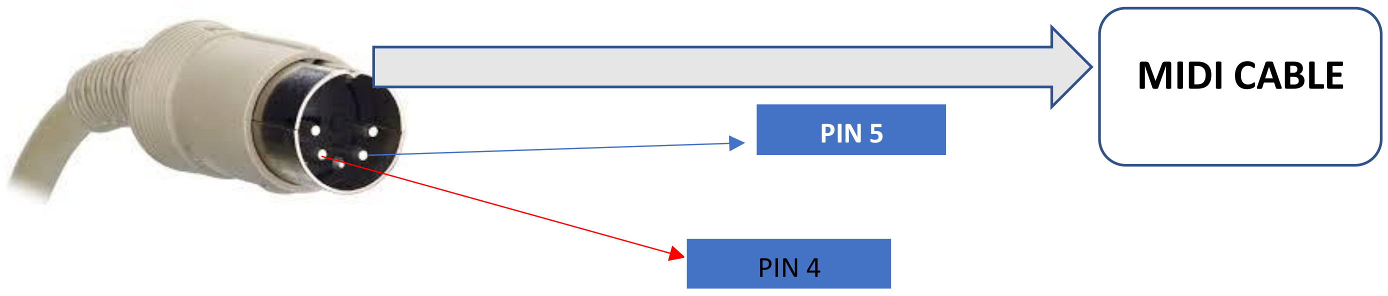
AND A MICROCONTROLLER THAT ALLOWS **SERIAL COMMUNICATION**

1) MIDI RECEIVER ARCHITECTURE:

The objective of the midi receiver is to receive the midi signal in input.
Below is the schematics for the midi receiver :

figure 1.0

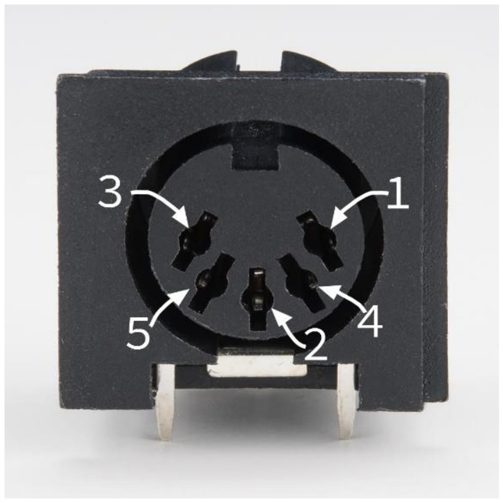


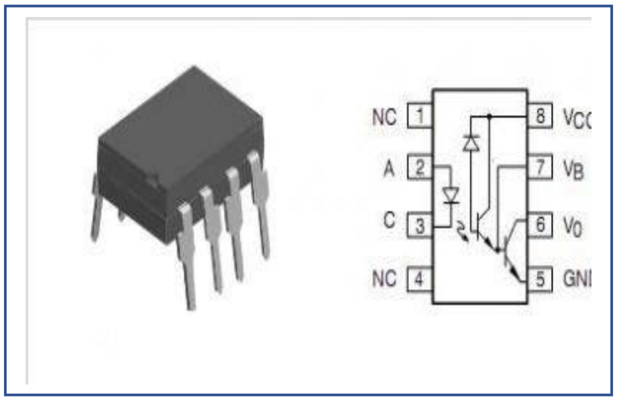


PROJECT PROPOSAL

Above is a male midi connector or simply a midi cable, pin 4 and pin 5 are the pins responsible for transmitting the midi signal. For the midi receiver, a **midi female connector** will be used to receive the midi signal from the midi keyboard via the midi cable.

MIDI FEMALE CONNECTOR ***6N138- OPTOCOUPLER***





As mentioned earlier the female midi connector receives the midi signal from the keyboard particularly in pins 4 and 5, but before the midi signal is transmitted to the microcontroller, it goes through an optoisolator or optocoupler, The optocoupler's task is to assure voltage isolation between the midi reception point and the microcontroller since Voltages greater than 5 volts can be damaging to the microcontroller.

<u>BRIEF INSIGHTS ON OPTOCOPULERS:</u>

Optocouplers are electronic component that transfers electrical signals between two isolated circuits, they are very common for transmitting digital data. The left side of the optocoupler is connected to the sending circuit while the right side of the optocoupler is connected to the receiving circuit therefore allowing isolation between two electrical communicating circuits.
Internally the left side of the optocoupler is comprised of a led, while the right side is comprised of a photosensor and a transistor that is activated when the photosensor senses some light from the led. The data sent is transcribed into the state of the transistor's collector that can be measured from one of the optocoupler's pins. So basically by measuring the voltage state (HIGH or LOW) of the transistor's collector we retrieve the data that was sent.

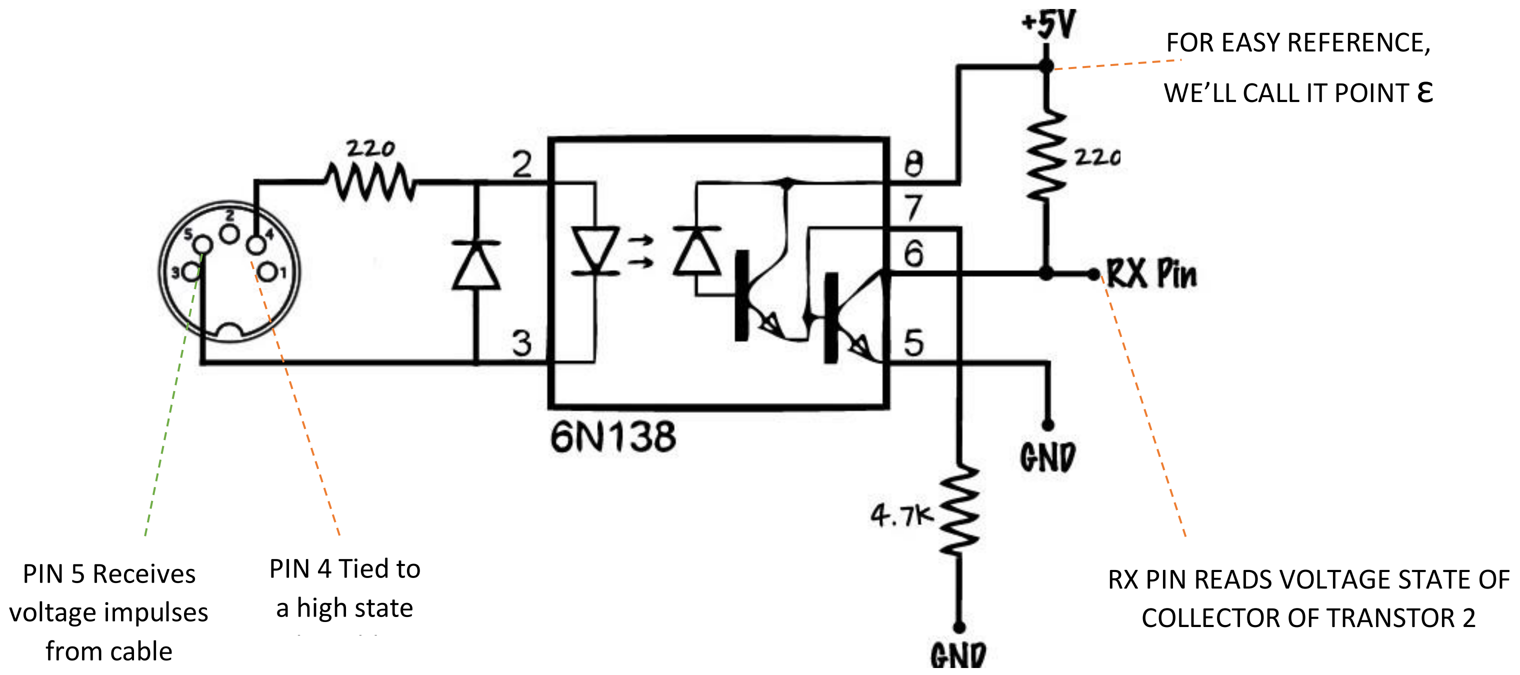
PROJECT PROPOSAL

HOW IS A MIDI SIGNAL SENT AND RECEIVED :

A midi device sends the midi data through pin 4 and 5 on the midi cable **by tying pin 4 to a high voltage state always** and sending voltage pulses through pin 5 *(by voltage pulses we intend either high or low state , 1 or 0 etc..)*, we should remember midi data is digital meaning they are just sequences of bits packed in group of bytes this will be explained further on.
The cable from the keyboard having its pin 4 in a high state ties **pin 4 on the midi female** connector to a high voltage and the pulses on pin 5 of the midi cable are transmitted to pin 5 of the female connector. Then the received bit is the **AND** operation between pin 4 and pin 5, meaning if pin 4 and pin 5 are both at high state, "1" is the bit received, if pin 5 is at a low state with pin 4 at a low state "0" is the bit received. And this is done at a rate of 31250 bits per second.

To correctly receive the midi data, an optocoupler is used to create voltage isolation since any high state above 5 voltage would damage the microcontroller.

Below is a clearer and more comprehensive schematic for figure 1.0, with the RX point indicating the point where the microcontroller collects the midi data.



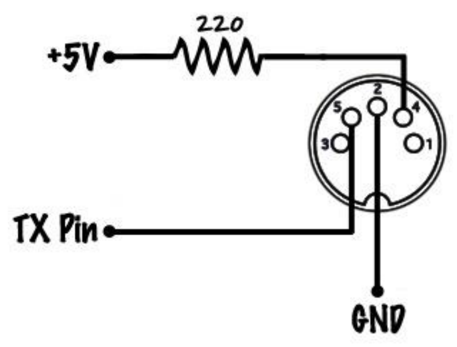
+5V
FOR EASY REFERENCE, WE'LL CALL IT POINT Ɛ
220
2
3
8
7
6
5
220
RX Pin
6N138
GND
4.7K
GND
PIN 5 Receives voltage impulses from cable
PIN 4 Tied to a high state
RX PIN READS VOLTAGE STATE OF COLLECTOR OF TRANSTOR 2

Pin 4 of the female midi connector is always at a high state meaning it claims the role of Vcc or voltage source. If the impulse on pin 5 is also at high state, then there is no current flow to the led inside the 6N138 Photo-Darlington Transistor Optocoupler since there is no voltage drop and the transistors inside the optocoupler are not activated **meaning there is no current flow from point ε** shown in the figure above, **therefore Point RX is at the same voltage of point ε.**

When pin 5 is at a low state, there is a voltage drop between pin 4 and PIN 5, therefore current flows to the led inside the 6N138 optocoupler which lights up to the photosensor that activates the transistors, the transistors in the optocoupler are activated, current flows from point **ε** to ground (the current goes from point **ε** to **collector of transistor 2** then to **emitter of transistor 2** then to ground). Due to the current path which necessarily implies a voltage drop across the resistor, point RX is definitely at 0 volt potential meaning the microcontroller will read bit “0” . In summary when a low state is sent at pin 5 through the midi cable to the female midi connector, the microcontroller detects a digital low state. When pin 5 is tied to a high state by the midi cable, the microcontroller reads a digital high state.

2) MIDI RECEIVER ARCHITECTURE:

The objective of the midi receiver is to transmit the midi signal in input.
Below is the schematics for the midi transmitter :



Pin 4 on the female midi connector is constantly tied to a high state (5 volts) by the microcontroller, then once there is information to send meaning the bass note has been computed by the microcontroller based on the notes received in input, the bytes that represents the bass note is then transmitted from the TX pin on the microcontroller to the female midi connector on pin 5. If a midi device is connected to the female midi connector through a midi cable, the bass note data is received and it is played by the midi device.

THE PROCESSOR:

To build the “PROCESSOR” which is the intermediary component which is the component between the transmitter and receiver, we need to understand the midi protocol. The midi protocol operates at a baud rate of 31250 bits per second and The MIDI language is used to transmit real time information (groups of bytes) . "Real time" means that each message is sent exactly at the moment and it must be interpreted by with no delay by the receiver.

So how are Midi messages or Midi events organized ?

Traditional MIDI messages are sent as a time sequence of 3 bytes (1byte = 8 bits). The first byte is a **STATUS byte**, often followed by 2 **DATA bytes**. A **STATUS byte** has bit 7 set to 1 and a **DATA byte** has bit 7 set to 0. The status byte can be further be decomposed into two nibbles (1 nibble = 4 bits).

STATUS BYTE = 1XXX**CCCC** with 1XXX = *note on* OR *off* message, CCCC = channel number

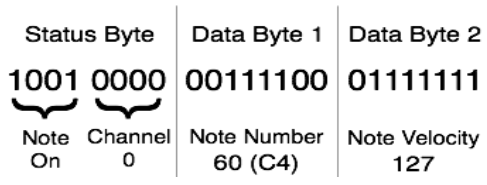
1st DATA BYTE =0YYYYYYY = Note number for example middle C_3 note number is 48

2nd DATA BYTE =0ZZZZZZ =Velocity number of the note, it describes how hard the note was pressed

The important point to know is that MIDI language does not define the sound itself, but only the sequence of instructions to create the sound in the receiver of the message**.**
The status byte has its most significant bit (most significant bit is the first bit from left) set to 1 always because the first nibble can only assume two values : note on or off
note on = 1001 which is 144 in decimal or 0F90 in hexadecimal
note off = 1000 which is 128 in decimal or 0F80 in hexadecimal
So a full status BYTE would of the form 1000CCCC or 1001CCCC.

Midi messages enable the transmission of only 128 types of notes pitches which approximate to 128/12 = 10 different octaves. This means the note values are in a range of (0-127) and can assume a maximum value of 127 and a minimum of 0. Since the maximum is 127 and 127 in binary is 01111111 meaning the most significant bit will always be set to 0, this also applies to 2nd data byte which represents the velocity information.

So a when a note is pressed the processor receives 3 bytes from the keyboard and when the note is released 3 further bytes are received.



PROJECT PROPOSAL

So when a note is pressed on the keyboard, RECEIVER receives 3 bytes and passes them to the microcontroller which is the PROCESSOR.
The bytes are : 1) status byte : **note on + channel number** 2) **the number of the note pressed** 3) **The velocity of the note.**

When the note is released 3 further bytes are received and they are: 1) **status byte : note off + channel number** 2) **the number of note pressed** 3) **The velocity of the note.**

So after the 3 components that comprise the object we are trying to build have been well defined and assembled together, Verification was sought by making the med-basser send everything it receives in input to another target synthesizer. This can be achieved by using a smartphone as the target receiver (a usb on the go cable would be needed) to receive any note played on the main keyboard. The code will be attached at the end of the report.