Energy Harvesting Based Self-Powered Wireless Plug and Play Keyboard

Jorge THIENPONDT, Jean-Pierre GOEMAERE, Lieven DE STRYCKER
Catholic University College Sint Lieven
Gebroeders Desmetstraat 1, B-9000 Ghent
info@dramco.be

***Abstract*— This paper presents a self-powered wireless pushbutton, which can be connected to a computer to be used as a Plug and Play keyboard. The receiver will be powered by an electromagnet power source. Every time a key is depressed an amount of energy of up to 130 µJ is generated. This is used to encode a unique ID of up to twelve bits and send this ID to a receiver. Due to the fact the transmitter is low-power it can send its code over a range of 5 m, even if the pushbutton was pressed rather gently. The receiver will decode this ID. The decoding algorithm can also decode non-ideal data packets, which originate from a pushbutton pressed to gently. Afterwards the decoded ID will be linked to a character and sent to the computer by means of a USB connection.**

***Index Terms*— Energy Harvesting, Keyboard, Self-Powered, USB, Wireless**

I. INTRODUCTION

During the last decades there has been a growing interest in self-powered wireless remote controllers, because hardwired interfaces are inflexible and wiring is expensive. On the other hand battery powered wireless devices need battery replacement and run flat on unwanted moments. The solution to this problem are energy harvesting based devices. For example the human body can be used as energy source to power small devices [1].

In [1] is calculated 130 gram of pressure is required to depress a key on a keyboard to type. This means 1,3 mJ of mechanical energy is required over a distance of 1 mm. Combining this fact with the information a skilled typist can write forty words a minute [4], an average power of 6,9 mW will be generated. Typing fast at a speed of ninety words a minute, generate up to 19 mW of power. This generated energy could be applied to supply the wireless transmission of the keyboard.

Papers [5]-[7] prove it is possible to make a self powered RF transmitter using a piezoelectric gas ignitor. In these cases a force of 15 N is required to send a twelve bit ID over a distance of 15 meter [5]. In this paper a similar aim is pursued. In this project a much lower force is on hand to actuate the pushbutton. Similar to [8] an electromagnetic approach to harvest energy is used in this project.

Zenith television from the 50's also had a wireless and batteryless self-powered remote control. Pressing the remote control actuates aluminum rods, which resonate on a fixed ultrasonic frequency. This sound will be decoded by the television and the volume or channel will be adjusted adequately. This system interferes with the pets and the settings of the television can be changed by surrounding noise. Another disadvantage of this system is that there is a need of a line-of-sight communication.

The last mentioned problem also arises using the well-known technology of infrared communication. To solve these problems radio frequency communication has been proposed as a solution.

To turn this whole project into a user-friendly application, a Plug and Play USB connection has been provided. The receiver emulates a USB keyboard. Every time a pushbutton is pushed, the corresponding code appears on the screen of the computer.

This paper will discus the transmitter and the receiver of the proposed wireless keyboard. The following chapter describes how the mechanical energy is converted into electrical energy. In the third chapter is stipulated how this voltage is regulated in the most efficient way to be used by the encoder and the transmitter, which are respectively described in the forth and the fifth chapter. In chapter six one can find the technical explanation of the receiver. This part goes deeper into the decoding algorithm at one hand and the USB protocol on the other hand. The paper concludes with an overview of the remarkable aspects and the possible improvements.

II. ENERGY SOURCE

To harvest the required energy from a push on a button an electromagnetic energy convertor is used. Therefore we modified a commercial available actuator of EnOcean, namely ECO100. Under normal circumstances a force of 2,5 N is required over a distance of 1,8 mm to actuate this bi-stable rocker. Every hit on the button makes a strong magnet swing back in a coil. This arouses a voltage over the windings of the coil. Using Faraday's law of induction, the induced voltage can be calculated using formula (1):

$$U_{ind} = -N\frac{\partial \phi}{\partial t} \tag{1}$$

N is the number of windings in the coil and ϕ represents the magnetic flux in one winding. Harvesting more energy involves the need for a fast change of the magnetic flux in the coil. The output voltage of a single push is shown in Fig. 1.

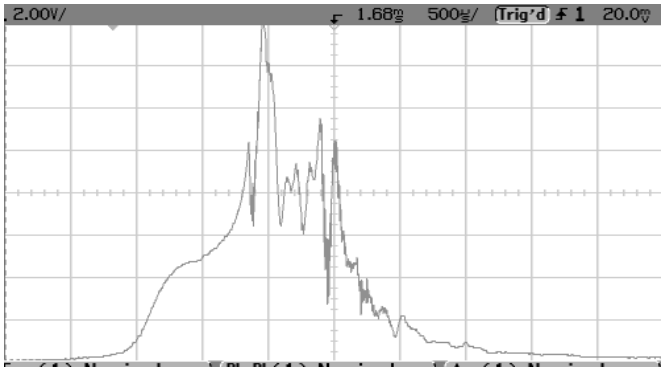

Fig. 1 Output voltage ECO100

This commercial available ECO100 is a bi-stable rocker, so it is not suitable to use it as a keyboard button. Some modifications have been done to covert it into a mono-stable switch to make it usable as a pushbutton. One strike on the button results in two pulses of the output voltage: pressing and releasing the button. As one can see in figure Fig. 3 a lever has been attached to reduce the force needed for one hit. The modified pushbutton now requires a force of less than 1 N on the tip of the lever over a distance of 6 mm.

Afterwards the output voltage is rectified using four Schotty diodes. The energy generated by the button is then stored in a storage capacitor.

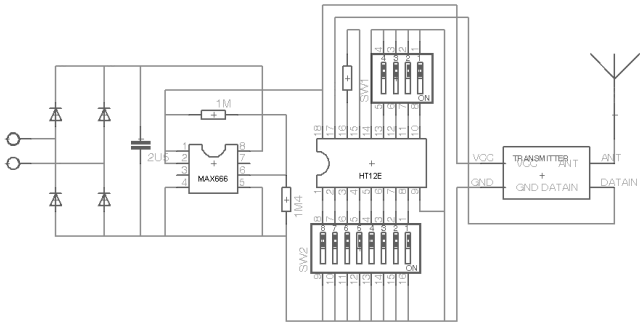

Fig. 2 Transmitter circuit

The capacity of this capacitor needs to be well dimensioned. A capacitor which is too small will not be able to store all the energy, so energy will get wasted. Choosing a capacitor too large is also not suitable. In this case the voltage over the capacitor will be too low to contain enough useful energy. Formula (2) gives the relationship between the capacity and the stored energy in the capacitor.

$$E = \frac{C\left(V_{High}^2 - V_{Low}^2\right)}{2} \qquad (2)$$

E represents the useful energy stored in the capacitor, with a capacity C and charged with a voltage of V_{High}. The energy that will get wasted is in this case determined by the minimum voltages required for the circuit used to send the data. This is represented by the value V_{Low}.

In this case a capacity of 2,5 µF suits best to contain the generated energy of 130 µJ.

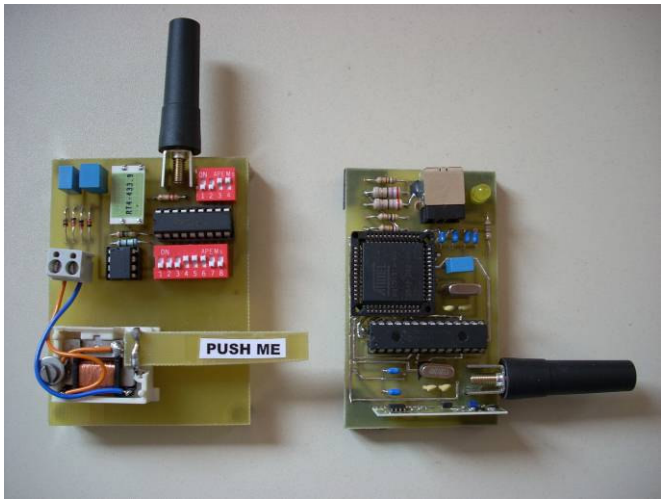

Fig. 3 Transmitter and receiver

III. VOLTAGE REGULATOR

This voltage, which varies between V_{High} and V_{Low}, needs to be stabilized to a voltage of 2,2 V. This is done by a linear voltage regulator MAX666. The resistors to feed back the output voltage have been raised higher than the recommended values of the datasheet, to reduce the power losses.

In [6] a switching voltage regulator is proposed to obtain a higher conversion efficiency. In spite of this advantage switching DC-DC convertors drain a higher quiescent power. In this case the entire circuit drains on average 444 mW using a linear voltage regulator MAX666, in spite of a switching DC-DC convertor NCP1400A, which uses 1836 mW under the same conditions. So it is preferable to use a linear voltage regulator for this purpose.

IV. DATA ENCODING

The main purpose of this keyboard is to broadcast data. Every key of the keyboard has a unique code. This code needs to be transmitted every time a button is stroked.

In this case we used a commercial available encoder HT12A of Holtek. This encoder converts four digital inputs of data and an address of eight bits into a serial stream of these twelve bits, which are preceded by a start pulse.

In this project the encoder will be used to encode twelve inputs, which represent the unique code of the pushbutton, into a pulse sequence. Every single input is encoded as a pulse. A low input is encoded with a low duty-cycle and a high input is encoded with a high duty-cycle. Every sequence of pulses is preceded by a short pulse equal to an encoded high input. An example of such a sequence is shown in Fig. 4. The lower trace shows a start pulse followed by the code “111000110011”. This code can be modified using twelve dip switches (see Fig. 3).

The sample rate used to encode the data can be set using a resistor. The transmitter uses the majority of the power dissipated by the circuit. Therefore a high conversion frequency would be desirable to reduce the overall power usage. But this encoding frequency has to be limited,

because the receiver is the bottleneck in this chain. There has to be made a trade-off between a high encoding frequency and the possibility to decode the received signals.

Another problem arises when the voltage over the capacitor gets significantly lower than the output voltage of the voltage regulator. In this case the voltage over the encoder deceases. This involves that the encoding frequency decreases at a fast pace. Because the data is encoded using pulse width, an encoded high input at the end of the data sequence can result in a broader pulse than an encoded low input at the beginning of the sequence. This situation occurs when the button is stroke to gently or at the end of the sent sequence. The voltage regulator has a low-voltage detecting function, which can be used to prevent sending those packages. To be able to send those data packages as well, we opted to solve this problem with a suiting decoding algorithm on the receiver's side.

V. TRANSMITTER

This code is sent to a transmitter AM-RT4-433 of RF Solutions to modulate this data sequence with a 433 MHz carrier, using OOK-modulation.

Fig. 4 shows that this transmitter uses most of the available energy. The upper trace represents the available energy in the capacitor and the second trace shows that two data packages are sent.

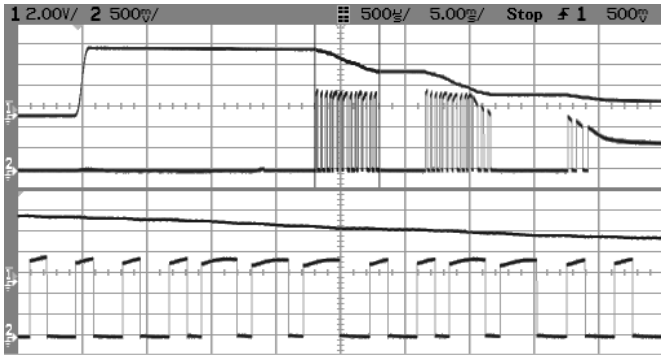

Fig. 4 Encoded data and energy usage + zoomed signals

VI. RECEIVER

The transmitted signals are captured using a receiver AM-HRR3-433 of RF Solutions. This receiver has an automated gain control, which makes it easy to filter noise signals when the actual code is received. The disadvantage of this feature is the fact that a lot of noised is received when the transmitter does not send any code.

To decode the received signals a microcontroller of Microchip is used, namely PIC18F2550. Every time a positive edge is detected, the microcontroller generates an interrupt to start a routine.

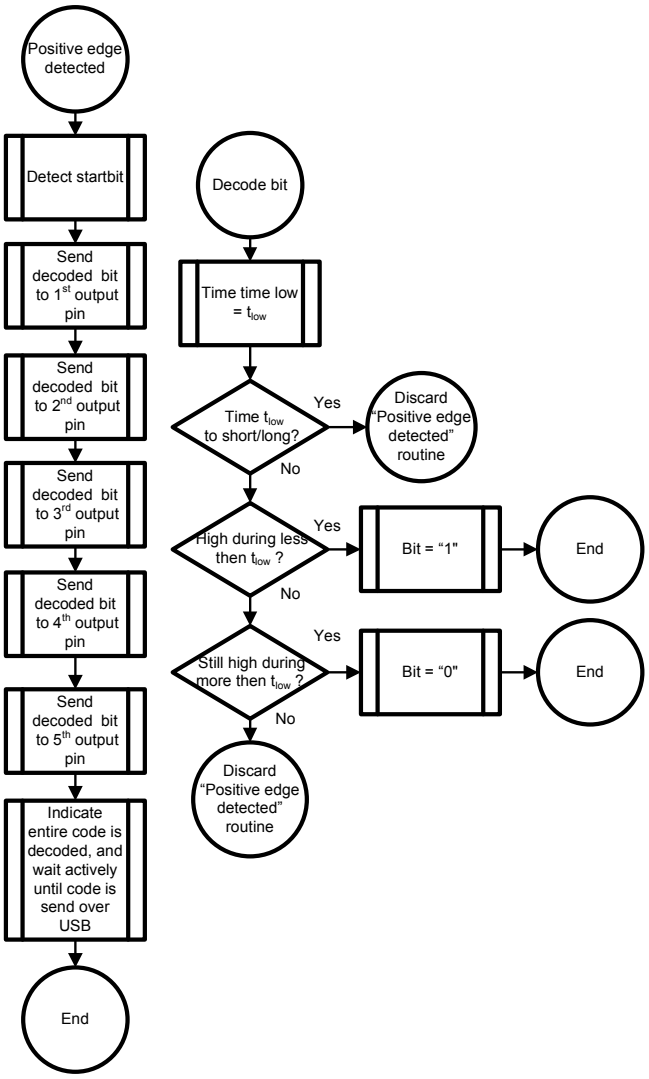

Fig. 5 Decoding algorithm

As one can see in Fig. 5, the microcontroller first measures the width of the start pulse. Afterwards it decodes only the first five bits of the twelve bit code. Using only these five bits a keyboard of up to 32 keys can be encoded. The advantage not to decode all the received bits is the fact that the pushbutton can be pushed more gently, because less energy will be needed to transmit the code.

As mentioned earlier, the data is coded using pulse width modulation. Therefore the time the signal is low is measured using a timer. If the signal is low between acceptable time limits, this could be a coded one or zero. If the signal is a shorter time high then it was low, this will be decoded as a one. And if it is longer high then double the time is was low, it concerns a zero. Any other pulse deviating from these conditions indicates a pulse not intended for this receiver. In this case the decoding algorithm is interrupted, until a new positive edge of the received signal is detected.

Using this algorithm gives good results to filter spikes and unwanted signals at one hand and it is also robust enough to decode data from a transmitter with a fading energy level at the other hand. In a practical use this algorithm doesn't get triggered by similar coded data, but sometimes it skips a valid code due to noise on the received signals. Fig. 6 shows

a case in which the transmitter sends four valid packages, but the receiver receives only two decodable packages. If the decoding is completed correctly, the decoded code is sent to another microcontroller that handles the USB protocol.

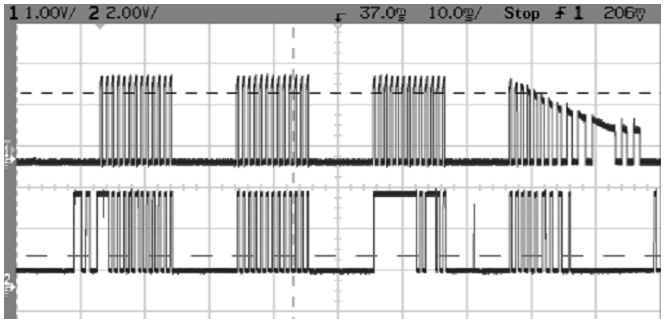

Fig. 6 Transmitted and received data

As mentioned earlier, the voltage regulator MAX666 on the transmitter has a rather low output voltage of 2,2 V. This implies the sending range of the transmitter is limited to 5 meter. This is still an acceptable range, taking into account the general purposes of a wireless keyboard. By adjusting the output voltage of the voltage regulator MAX666 the range of the transmitter can be adjusted. Under normal circumstances the transmitter has a range of 70 meter. To send one package of data under normal circumstances it consumes 20 ms . 2,4 V . 1,03 mA = 50 µJ of energy. If only a voltage of 1,2 V is used, the current will reduce and only 20 ms . 1,2 V . 420 µA = 10 µJ of energy will be required. But in the last case the transmitting range will be reduced down to 30 cm. In this project a tradeoff is made between transmitting range and energy consumption to obtain functional characteristics.

To make the receiver user friendly, a USB 1.1 functionality has been implemented. The user will only need to plug the receiver into his/her computer to use the keyboard.

The USB protocol is built from a layered structure. This way one does not have to know the details of the lower or higher level, when implementing something in a layer.

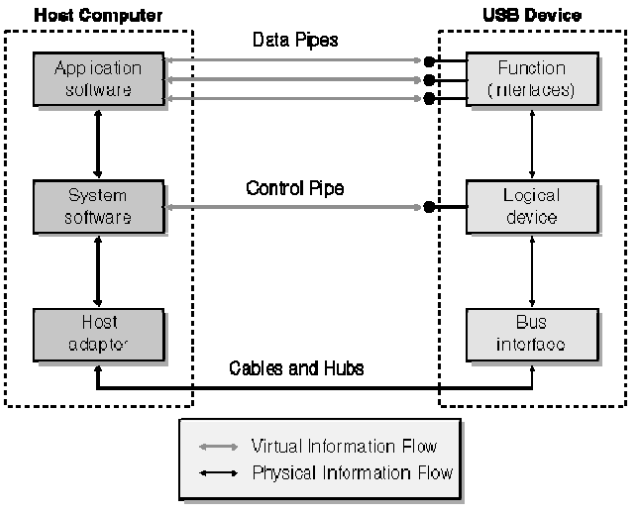

Fig. 7 Layered structure of USB

At the bottom of the structure is the hardware layer. This is called the USB Bus Interface Layer. Those are the physical signals, ones and the zeros, bits and bytes, which are sent over the USB cable. The upper layer, the Function Layer contains the user program. Here the keyboard will send its data toward the PC as an interrupt transfer. An intermediate layer links the software layer and the hardware layer. This is the USB Device Layer, from which we retrieve the drivers. In this layer the control communication takes place during the enumeration process and here the host computer will poll if new data has been received.

In a contemporary operating system there are some built-in drivers. These are known as Class Specific drivers. Plug and Play at USB is a term that you will already have heard of at least once. When a Plug and Play USB device is connected, it will automatically be installed and be ready for use without the user having to install drivers or changing settings. In this case a Class Specific driver was loaded. This enumeration process of this project is demonstrated in Fig.8.

Fig. 8 USB Plug-and-Play device

We also encounter Class Specific drivers available that are more specifically intended for USB Keyboards. We have focused on these drivers to design a small USB keyboard. This way the receiver will work on any PC whether the user prefers Windows 98, 2000, XP, Vista or even a Linux distribution. Because of the Hot Plugging and Plug and Play features, all the user has to do is connect the receiver by means of USB.

To be able to use the Class Specific driver this must be described in the device. The entire description of the device is stored in so-called Descriptors.

Those descriptors determine how much current the device can use, and of course they describe it concerns a keyboard. These descriptions are very complete and vast. For example the format, the language, … of keyboard have to be described.

The descriptors will also be stored in a layered structure. This way we can distinguish the “Device Descriptor” which gives a general description of the device. From this descriptor one can retrieve for example the name of the product and vendor, or the supported USB version.

Below this layer the “Configuration Descriptor” can be found. All provisions concerning energy usage are stipulated closely here.

The underlying layer exists of an “Interface Descriptor”,

used to give a description of the structure and functionalities of the device. This is the header of the features of USB device.

A following descriptor is the "HID (Human Interface Devices) Descriptor" and the "Report Descriptors" which determine the specifications of the physical layout of the keys on the keyboards (e.g. QWERTY), and the type of application where it is used for. In this project it is a replacement for a well known notepad with a Belgian country code.

A last important descriptor used, is the "Endpoint Descriptor" where one can find detailed information on how the transferred data will be encoded.

These are the necessary descriptors which make it possible to create a self installing device. These and other descriptors are stored in the device. The computer is the host and will regulate all transfer from and toward the USB device. But the computer does not yet know what the connected device looks like. For this reason the computer will send requests to the device. Requests can be used to make the device send a certain descriptor to the host. These requests can also be used for numerous other purposes, for example polling if data has been received from the wireless keyboard. As mentioned earlier, within the USB protocol the computer is the host. This means that every data transfer over USB will be introduced by a request. So when the PC wishes to read in the received character, a request will first be sent to the device. As an answer to this request the device (the receiver of the keyboard) will respond if data has been received.

To handle this USB protocol an 8051-based microcontroller is used. More specific the AT89C5131 of Atmel (Fig. 9). This microcontroller has 32 Kbyte on-board Flash memory, and six programmable endpoint, from which only two are used: the FIFO Endpoint 0, used for the Control Transfer, and the 32 byte IN Endpoint 1 to send the received character to the PC.

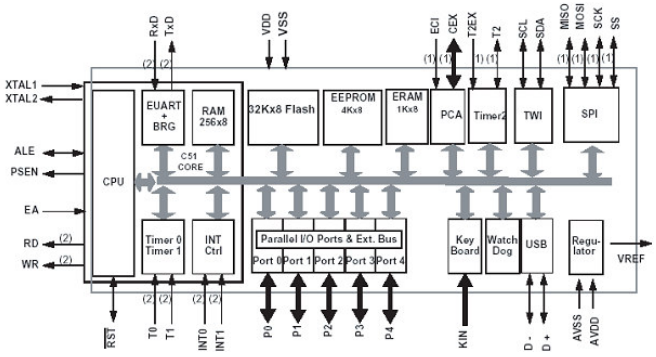

Fig. 9 AT89C5131 microcontroller

VII. FUTURE WORK

Although this prototype has satisfying results, a lot of improvements can be made.

To generalize this transmitter into a keyboard with multiple keys a lot of redundant components will be incorporated. In the current setup every pushbutton has its own electromagnetic energy source, its own voltage regulator, its own encoder and its own transmitter. To make it space and cost efficient a keyboard only needs one transmitter and one encoder and as few as possible electromagnetic energy sources. Some of the generated energy can be spent to power a multiplexing circuit.

The encoder can be replaced by a low-power microcontroller. Doing this makes it possible to reduce or raise the number of encoded bits in one data packet. Also some energy can be saved by dropping the first synchronizing start pulse. As compensation the decoding algorithm will need to be adapted.

As one can see in Fig. 10 and Fig. 11 the transmitter requires more energy to send a "0" than a "1", due to the fact that the transmitter will be turned on approximately three times longer to send a "1" than a "0". As a solution to this problem another algorithm to encode the data can be used in combination with a low-power microcontroller, with a critical approach on the energy consumption of the transmitter and on encoding "1" and "0" equally in terms of power consumption.

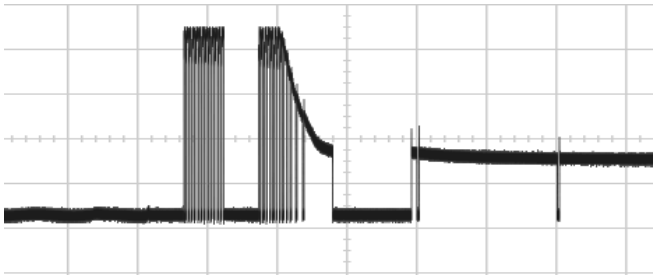

Fig. 10 Sending code "000000000000"

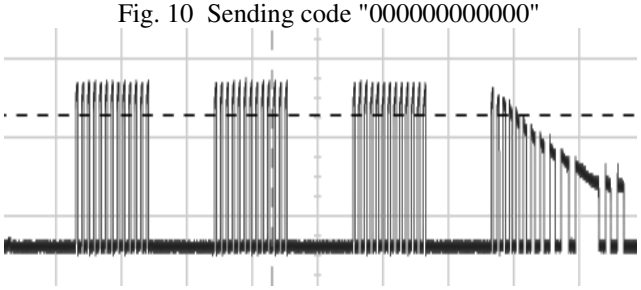

Fig. 11 Sending code "111111111111"

A main restriction of reducing the power consumption of the transmitter can be found at the receiver's side. The used receiver can not adequately replicate signals which are sent at higher frequencies. Without this bottleneck it would be possible to encode the data at a higher frequency which has as a consequence less energy will get wasted on the transmitting of the data. These results in either a higher reliability of the transmission or either in a lower force needed to actuate a pushbutton.

To make the receiver cost efficient and reduce space, the microcontroller PIC18F2550 used to decode the received signals, and the microcontroller AT89C5131, which handles the USB protocol, can be replaced by on of those two microcontrollers.

VIII. CONCLUSION

In this paper a successful implementation of a wireless pushbutton has been presented. The system exists of a transmitter and a receiver.

The transmitter consists of an electromagnetic pushbutton which makes it self-powered. Every time the button is pressed it transmits a code of up to twelve bits.

The receiver decodes the received signal. Therefore a robust algorithm has been implemented that can decode an imperfect data packet. This Plug and Play keyboard can be connected to a computer by means of a USB connection.

This prototype is acceptable to be used as a wireless keyboard. It can send the twelve bit code over a transmission range of up to 5 meter with an amount of energy of less then 130 µJ.

But there is still a lot of improvement possible. Mainly the receiver components are the bottleneck of this system and the encoding of the data can be improved using a microcontroller instead of a commercial encoder.

REFERENCES

[1] N.S. Schenck and J.A. Paradiso, "Energy Scavenging with Shoe-Mounted Piezoelectrics", IEEE Micro, vol.21, pp.30-41, 2001.

[2] S. Roundy, B. P. Otis, Y. H. Chee, J. M. Rabaey, P. Wright, "A 1.9GHz RF Transmit Beacon using Environmentally Scavenged Energy", ISPLED 2003, Seoul Korea , 2003.

[3] T. Starner. The cyborgs are coming, Technical Report 318, Perceptual Computing, MIT Media Laboratory, 1994.

[4] K. Lyons, T. Starner, D. Plaisted, J. Fusia, A. Lyons, A. Drew, and E. Looney, "Twiddler typing: One-handed chording text entry for mobile phones", Human Factors in Computing Systems (CHI 2004 proceedings), pp. 671 – 678, Vienna, Austria, April 27-29 2004.

[5] J. A. Paradiso and M. Feldmeier, "A Compact, Wireless, Self-Powered Pushbutton Controller", Ubicomp 2001: Ubiquitous Computing," ACM UBICOMP Conference Proceedings, Atlanta GA, 2001, Springer-Verlag Berlin Heidelberg, 2001, MIT Media Laboratory, 2002.

[6] Y.K. Tan, K.Y. Hoe, and S.K. Panda, "Energy Harvesting using Piezoelectric Igniter for Self-Powered Radio Frequency (RF) Wireless Sensors", Industrial Technology, IEEE International Conference on Volume, 2006.

[7] K. Y. Hoe, "An Investigation of Self Powered RF Wireless Sensors", National University of Singapore, 2006.

[8] Crisan, A., Typing Power, US Patent No. 5,911,529, June 15, 1999.