SDR Implementation for DCF77

Filip Zaplata, Miroslav Kasal
Dept. of Radio Electronics
Brno University of Technology
Purkynova 118, 612 00 Brno, Czech Republic
xzapla00@stud.feec.vutbr.cz, kasal@feec.vutbr.cz

Abstract— **The DCF77 signal is used to carry out the information of precise time for long distances. Standard DCF77 receivers consist of AM long wave receiver circuit, connected to a data processing unit. Whole AM receiver circuit is often integrated into a chip with some external components. Such circuit gives demodulated logical data output, which can be led right to the processor to be decoded and presented. The idea is to remove the analog demodulator and process the amplified RF signal also on the processor. Specialized AM radio chip then would be replaced by a simple amplifier, e.g. operational amplifier circuit. Nowadays, microcontrollers include AD converters fast enough to digitalize signals at frequencies above 77.5 kHz. Using the advantages of undersampling, Goertzel algorithm and correlation techniques, it allows to make program very low computational power demanding. Developed algorithm is supposed to be implemented to 70 MHz ARM7 LPC2103 microcontroller. Examples of code realization in GCC are introduced also with discussion of possible complications.**

Keywords— DCF77; time stamp transmission; software defined radio; digital receiver; Goertzel algorithm; mathecd filter;

I. INTRODUCTION

One of the precise time sources can be the radio frequency signal carrying the information – the time stamp. In the world there are many transmitters of which only purpose is to be the source of time stamp for its relatively close vicinity. In the age of the positioning systems like the GPS time stamp transmitters recede to the background, but their low frequency carrier signal keeps them important for the simplest systems like an alarm clock or a street clock. On the other hand, since the generation of the signal inside the transmitter is based on very precise atomic clocks same as in positioning systems, the accuracy of the receivers could be the same. In Europe the most known time stamp transmitter is probably the DCF77.

Due to the low carrier frequency, below 100 kHz, and amplitude shift keying modulation used in almost all systems, the receiver could be very simple. On the market it is possible to find several integrated receivers. The older ones, nowadays hard to get, can be hard-wired to receive the signal on selected frequency [1] or [2]. Still available may be the more improved receivers [3] and [4], which in addition support switching frequencies or include system type recognition and a decoder. But all these solutions are based on the tuned frequency receiver with quartz filter and possibly diode detector. The demodulated signal is then digitalized by a comparator and preprocessed by a logic or simply put on the output. It is obvious that the digital signal has to be processed substantially more to get desired data. Also presentation could consume some processing effort.

The software defined radio architecture deals with the idea to move the AD conversion right behind to the antenna in general receiver block diagram. Moving the AD converter (the comparator) closer to the antenna may lead to move the demodulator and decoder from analog circuit into a single processing unit together with the data processing application. The antenna and selective low noise amplifier remain analog circuits feeding the AD converter. A bit resolution of the converter then has to be higher than 1 bit of a comparator, also its speed has to be high enough to be able to process signals of higher frequencies than the time stamp system carrier. However such AD converter is included as a peripheral in most modern microcontrollers. The effective system of algorithms designed for the implementation into the LPC2103 microcontroller will be discussed in the following.

II. DCF77 STANDARD

As [5] says, the DCF77 transmitter is located 25 km far from Frankfurt/Main in Germany. Transmission began on 1 January 1959. Physikalisch-Technische Bundesanstalt (PTB) was involved in the broadcasting from the beginning and today PTB is solely responsible for it. The transmitter works on the carrier of frequency 77.5 kHz derived from PTBs atomic clock. Its deviation is on average less than relatively $2 \cdot 10^{-12}$ on one day and on average less than relatively $2 \cdot 10^{-13}$ during 100 days from the nominal value. There are phase time deviations about 5.5 ± 3 µs caused by the pseudorandom phase modulation which has to be taken into account. However over long measuring time the phase time variations average out to approximately 0.1 µs and can be disregarded. The station uses 50 kW semiconductor transmitter and vertical omnidirectional antenna 150 m high. The EIRP is approximately 30 kW. Due to the low frequency the signal propagates mainly as a ground wave along the Earth's surface and lesser part is transmitted to the sky and can reflect on the ionospheric D layer. That is why the reach is around 2000 km. Up to distances below 500 km the field strength can be expected of 1 mV/m.

The time data is modulated on the carrier by an amplitude shift keying. At the beginning of each second the amplitude is reduced to approximately 15% for the duration of 100 ms or 200 ms meaning binary 0 or 1 respectively. An example of the signal envelope is shown in Figure 1. Reduction is phase-synchronous with the carrier oscillation and its falling edge marks the precise beginning of the second. Thus the symbols

978-1-4673-5517-9

are called second marks. Each second mark carries information of one bit, the last second mark of minute is left out, and this is the information about the end of the minute and also the data frame.

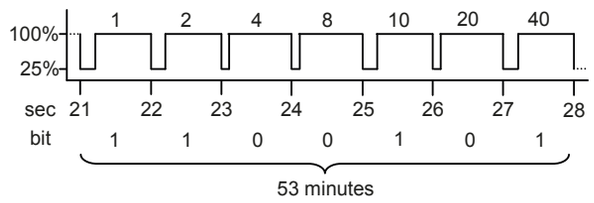

Figure 1. Modulation signal example [8].

It means that the whole data frame is 59 bit long and is carried within each minute. The data frame contains the information about time actual right after the reception of the minute mark. Figure 2 shows the format in which data are arranged in the frame. The first 14 bits are probably still used by a third party for transmitting warning information, but this data are usually omitted by receivers.

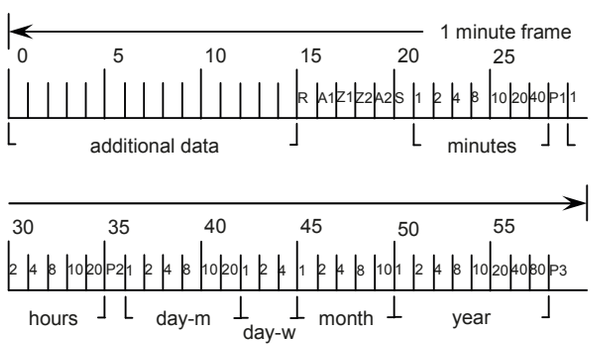

Figure 2. DCF77 data description [8].

Additionally carrier is phase modulated with the pseudorandom sequence of ±13°. The mean value of the carrier's phase remains unchanged, but receiving of the sequence by cross-correlation allows to determine the time stamp positions more precisely. This feature is omitted in the following discussion. More information can be found in [5].

III. AD CONVERSION

The Nyquist sampling is well-known technique used for digitalization of relatively low frequency signals. On the other hand most RF signals occupy only relatively small range of frequencies, thus are bandpass signals. Aliasing will degrade the signal if it occurs when Nyquist sampling is used, that is why the sharp lowpass anti-aliasing filter has to be used. Aliasing is in fact the transfer of the band from higher frequency segments (higher Nyquist zones) to base band (the first Nyquist zone). Nyquist zones are the bands into which the frequency is divided by the sampling and the half of the sampling frequency is its bandwidth.

Actually the transfer can be provided without any degradation of the aliased signal. The condition is that the frequencies out of the desired band has to be suppressed, in other words instead of low pass filter the sharp bandpass filter has to be used. Such technique is known as bandpass sampling and is often used for AD conversion of RF signals also due to its downconversion ability. Of course the realization is not so simple as it seems to be and suffers from more effects than Nyquist sampling does. The detailed analysis is not the aim of this paper, but it can be found in [6].

DCF77 signal at carrier of 77.5 kHz and about 2.5 kHz bandwidth seems to be a good candidate for bandpass sampling application. The basic requirements on the AD converter are the ability to sample signals of frequencies as high as the DCF77 signal contains and sufficient resolution to guarantee high dynamic range. Fortunately such converters are nowadays available on many common microcontrollers as its peripheral. The LPC2103 microcontroller contains 10-bit AD converter able to convert signals up to 400 kHz. Figure 3 shows the chosen frequency plan. The converter runs at 24 kHz sampling frequency, the carrier from 7th Nyquist zone is then downconverted to the 5.5 kHz at the base band. Plan supports theoretically bandwidth up to 12 kHz, which is enough to support the bandwidth of DCF77 and a low suppression of unwanted frequencies. To set the correct frequency of the AD converter, it is necessary to set some registers and choose the right frequency of the CPU clock. Equation (1) shows the peripheral clock frequency *PBCLK* computation from the main crystal frequency f_{osc}. The *MSEL* register sets the multiplier of the phase locked loop and *APBDIV* is a peripheral clock divider. The clock of the AD converter is derived directly from the peripheral clock by the divider *ADCDIV* (2). Where f_S stands for desired sampling frequency and *CLKS* is the register of the resolution. The resolution of the AD converter is chosen to be 9-bit, it means theoretically about 55 dB dynamic range. Such low dynamic range will work for testing purposes, otherwise it should be improved by an AGC amplifier. Table I summarizes computed parameters for this example case.

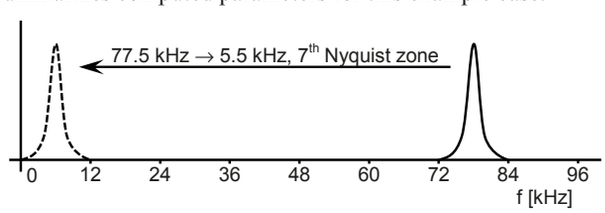

Figure 3. Bandpass sampling frequency plan [8].

TABLE I. LPC2103 AD CONVERTER SETTINGS.

Parameter	Value
MSEL	5
APBCLK	4
f_{osc}	12288000 Hz
PBCLK	15360000 Hz
CLKS	9
f_S	24000 Hz
ADCDIV	64

$$PBCLK = \frac{MSEL}{APBDIV} f_{osc} \quad (1)$$

$$ADCDIV = \frac{PBCLK}{f_S \cdot CLKS} \quad (2)$$

IV. SIGNAL PROCESSING

The functional diagram of designed software is shown in Figure 4. Firstly the desired band is selected by the selective bandpass filter. For such purpose the Goertzel algorithm is used. This decision brings also the advantage of the decimation and power computation in a highly effective algorithm. The output of the filter still may contain disturbances, so the signal is slightly averaged to suppress the sharpest peaks. It also helps to prevent overflowing in the following processing steps. Averaged power envelope can already be converted to a binary representation by a comparator as it is done by analog integrated circuits, but the noise included may be still so strong, that it will produce many miss-detections and the data will be corrupted. Due to simple pulse modulation and periodicity, the usage of correlation seems to be wise. Even the implementation is due to the simple correlation pattern very effective. The correlator gives correlation peaks as its output and these are processed in peak detectors. There are two peak detectors, one for falling edge of the envelope i.e. the positive peak, and the second one for rising edge i.e. negative peak, but both works in the same manner. Peak detector at first compares the input signal with the crop level, which is simply acquired by a long term averaging of the correlator output, and the levels under the crop level are cropped out – set to zero. The remainder is the signal containing still fluently varying peaks with variable width. The second step of the peak detector is then to find the maximal value of the peak and the left over values are also set to zero. These resulting spikes are already usable to synchronize the receiver and detect data. By the correlator the signal was split into the rising edge branch and the falling edge branch. Because the falling edge is the precise information about the time of the start of second, it is wise to use that peaks for synchronization of the second counter. Afterwards it is easy to measure the time from synchronization mark to time of the rising edge peak and use that time difference to decide which bit was probably sent. The second lock block also can detect the loss of the peak, which means that the signal is corrupted or the minute synchronization symbol was sent. Minute lock block processes these events and synchronizes its counter when the time space between losses is precisely 60 seconds. These outputs can already be used for successful decoding of the received data.

A. Goertzel algorithm implementation

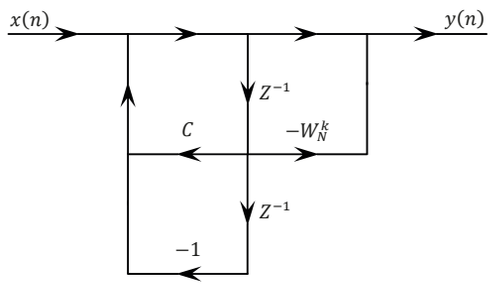

Figure 5. Signal diagram of Goertzel algorithm [8].

$$H(z) = \frac{z}{z - e^{j2\pi\frac{k}{N}}} \quad (3)$$

The Goertzel algorithm is based on a discrete Fourier transform, but computes the complex value only of the selected spectral position (bin). The general transfer function (3) [7] derived right from the discrete Fourier transform contains complex coefficient, therefore this form is not very effective. By the expansion the equation can be modified and then split into a form ready to be implemented (4). For short there is used the coefficient C, by which the desired bin is selected as is in (5). The selection of the bin may be described in two ways, by the Goertzel filter length N, which directly corresponds with the length of Fourier transform, and bin index k, or by the normalized frequency f/f_S, where f_S is a sampling frequency and f is a frequency of desired bin. The difference equations have a form of (6) and (7), where n is an index of a sample and x is the input sample, and corresponding signal diagram is shown in Figure 5. The length of the algorithm directly affects its frequency resolution the same way as the number of input samples affects the resolution of Fourier transform, in other words the longer the algorithm the more selective the filter. Particularly the length of the filter N means the number of

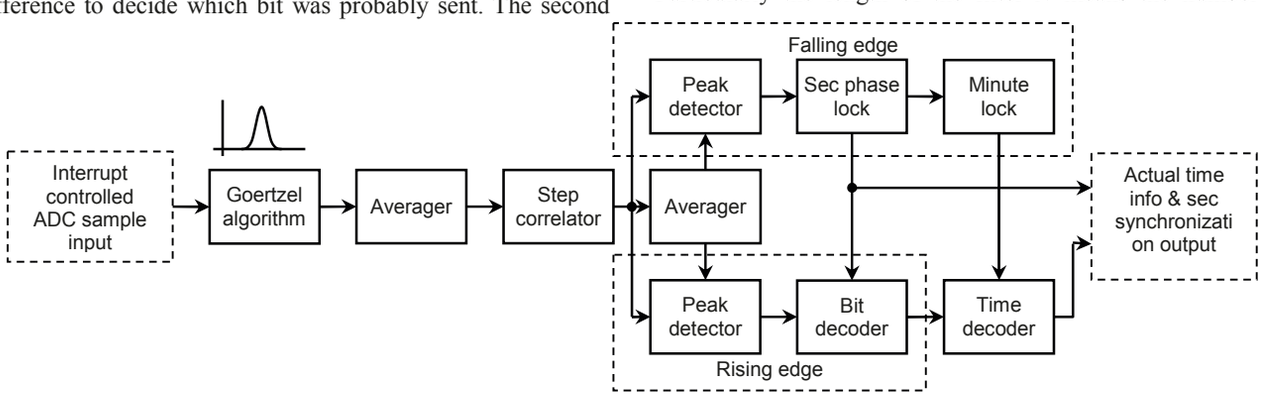

Figure 4. Block schematic of realization [8].

iterations of the algorithm i.e. the length of the convolution. After every *N* iterations the result is valid and before the next start all buffers has to be cleared, this is where the *N*-decimation ability comes from.

$$H(z) = \frac{z^2}{z^2 - 2zcos\left(2\pi\frac{k}{N}\right) + 1} \cdot \frac{z - e^{-j2\pi\frac{k}{N}}}{z} \tag{4}$$

$$C = 2\cos\left(2\pi\frac{k}{N}\right) = 2\cos\left(2\pi\frac{f}{f_S}\right) \tag{5}$$

$$y(n+2) = x(n+2) + Cy(n+1) - y(n) \tag{6}$$

$$y(n+1) = x(n+1) - x(n)e^{-j2\pi\frac{k}{N}} \tag{7}$$

The second part of the filter is non-recursive, that is why the evaluation can be used only if the result is needed, and it is right after *N* iterations. The result of the final computation is the complex value of the frequency component. For the purpose of the signal demodulation it is necessary to know the absolute value of that sample, it is calculated by a powering the complex value (8). It is not necessary to follow the strict rule for the computation of absolute value from the complex number by evaluating a square-root, the power is even more helpful.

$$|y(n+1)| = \sqrt{x^2(n+1) - Cx(n+1)x(n) + x^2(n)} \tag{8}$$

```
signed short sample;
signed int sk;
static signed int d1 = 0, d2 = 0;
static unsigned short term_cnt = 0;

sample = (ADGDR>>6)&0x3ff;

sk = (((((sample-d2)<<GCOEF_ACCURACY)+(COEF*d1))
     >>(GCOEF_ACCURACY-1))+1)>>1;
d2=d1;
d1=sk;

if (++term_cnt == GOER_TERM)
{
   term_cnt = 0;
   sk = (((d1*d1+d2*d2-((((COEF*d1)>>4)*d2)
        >>10))>>5)+1)>>1;
   d1 = 0;
   d2 = 0;
   *adc_buff_cur++ = (unsigned short)sk;
   record_time = timercounter;
}
```

recursive part of the algorithm

rounding

non-recursive part

Figure 6. Goertzel algorithm code.

The example of the implementation of Goertzel algorithm is shown in Figure 6, it is the part of code of interrupt service routine. Other parts of code are not so critical due to Goertzel's decimation ability, that is why it can be serviced within the interrupt routine. Note that all computations include simple rounding code to diminish the error. When *N* iterations are done (the counter reaches *GOER_TERM* constant), the power evaluation will be initiated. Additionally each power sample is marked with the time stamp of its acquisition.

Both iterative evaluation and power evaluation are the most critical issues on overflowing. Since the algorithm is directly derived from discrete Fourier transform the result of the evaluation of the convolution will be *N*-times higher than the input. Such restriction has to be encountered also with respect to the fixed point representation (*GCOEF_ACCURACY*) of the Goertzel coefficient to prevent overflowing.

B. Correlator

The DCF77 base band signal is a pulse modulation envelope of only two keys coding binary zero and binary one. Due to its simplicity the signal is possible to be decoded by a matched filter. Matched filter uses the correlation's capability to find a special pattern in the noisy signal. When the pattern matches the output of the correlation will suddenly rise in a well-noticeable peak.

$$CL = 2 \cdot \frac{f_S \cdot t_{inv}}{GL} \cdot RF \tag{9}$$

The correlation pattern is derived from the length of the shortest invariant part of the symbol, i.e. 100 ms for binary zero. To be applicable to both symbols the pattern is a form of a step function, then it will correlate with a similar step of input signal with the result of peaks at the same position as the edges are. Depending on the polarity of the peak it is possible to decide whether the falling or rising edge has come. The pattern is of length (*CL*) twice of the invariant part length t_{inv} as shows equation (9). To define the time, sampling frequency f_S is needed. Of course the decimation factor of the Goertzel filter has to be considered, the *GL* variable stands for the Goertzel length and is equal to *N*. *RL* is a reduction factor, which optimizes the result for non-ideally sharp edges of input signal. Its value is in ideal conditions equal to 1, but practical value is between 0.9 and 1. Suitable value is found experimentally.

```
static unsigned short delayed[STEP_LEN];
static unsigned short cursor=0;
unsigned short i, j, k;
signed int output;

delayed[cursor++]=input;
if (cursor>=STEP_LEN) cursor=0;
j=cursor;
k=cursor+(STEP_LEN>>1);
if (k>=STEP_LEN) k-=STEP_LEN;
output=0;

for (i=0; i<(STEP_LEN>>1); i++)
{
   output+=delayed[j++];
   output-=delayed[k++];
   if (j>=STEP_LEN) j=0;
   if (k>=STEP_LEN) k=0;
}
```

storing input sample

pointer initialization

step correlation by alternating addition and subtraction

Figure 7. Correlator code.

Figure 7 shows the sample code of correlator. It is obvious that the correlation with the step pattern is performed in a very simple way, the input sample is first stored in a circular buffer

and then one half is added to the *output* variable while the second one is subtracted. This is very effective algorithm and since each sample is processed separately with alternating operation (sign) the possibility of overflowing is diminished.

In the situation, when the correlation fits precisely, the output peak will be *CL*-times higher than the input sample. It brings the ability of correlation to improve the signal to noise ration also *CL*-times. The *output* variable has to be able to hold such amount of data.

C. Peak detection

To define precise time position of the peak it is necessary to have a peak of one sample width, but the output of the correlator rises and falls fluently through several samples. Fortunately the peak usually has its own single maximum at the precise position of the edge, therefore the issue is to separate the maximum value of the peak. The additional noise may cause several adjacent peaks of significantly lower height. Thus the signal is first cropped by twice a level of a long term average of the correlation signal, and then separated high peaks can be led to the maximum detector. The code of the peak detector algorithm is shown in Figure 8.

```
static unsigned int m = 0, d = 0, out = 0, dout = 0;

dout=out;

if (input < crop) { out = 0; m = 0; }
else
{
   out = input;
   if (input < d) out = 0;
   else
   {
        if (input > m) m = input;
        dout = 0;
   }
   d = input;
}
```

cropper

maximum detector

Figure 8. Peak detector code.

The maximum detector simply stores the incoming sample and when the next sample comes, samples are compared. When the stored sample is bigger, its value is returned, otherwise the returned value is zero. It is obvious that the returned value is one sample delayed and the algorithm is able to find only the local maximum. If some fluctuations within the peak appear it will cause an output looking like a burst of spikes around the peak position causing the deviation from the right position.

D. Signal averaging

In the system there are used two averagers, first to decrease the deviation of the Goertzel filter output and the second one to find the crop level for the peak detector. Although one is a long term average both use the same algorithm, only their coefficients are different. Exponential moving average is a very simple algorithm and still gives sufficient results, therefore it is a good candidate to be implemented. Figure 9 shows the signal diagram and (10) is the difference equation, where *n* is an index of the sample and *x* is an input signal, *C* is an average coefficient selected from the interval $0 \div 1$. The output value has to be weighted by a coefficient *1-C* to give unified output. Weighting coefficient is derived from absolute value of the transfer function (11) as shows (12).

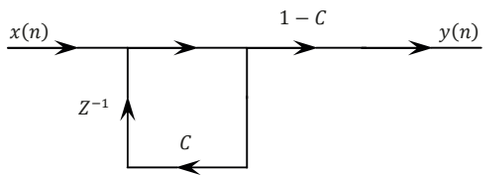

Figure 9. Signal diagram of average [8].

$$y(n+1) = x(n+1) + Cy(n) \tag{10}$$

$$H(z) = \frac{z}{z - C} \tag{11}$$

Code of realization is shown in Figure 10. Without a proof the maximum of transfer function is when *ω=0* and is dependent on the coefficient (12). The variables have to be optimized to handle the values to prevent overflowing.

$$\left|H(e^{j\omega})\right| = \frac{1}{\sqrt{1 - 2Ccos(\omega) + C^2}} \xrightarrow{\omega=0} \frac{1}{1 - C} \tag{12}$$

```
static signed int d=0;
signed int output;

output=input+d;
d=(((coef*output)>>(ACOEF_ACCURACY-1))+1)>>1;

output=(((output*((1<<ACOEF_ACCURACY)-coef))
   >>(ACOEF_ACCURACY-1))+1)>>1;
```

rounding

output weighting

Figure 10. Averager code.

E. Locking to time marks

The chain of processing blocks at this position gives the information about occurrence of the edge, its slope and the time of acquisition. The falling edges, marked as a positive value of pulse inform about the start of each second and therefore are used for second synchronization. The negative pulse is then used to compare its time of occurrence with the start of a second given by synchronized second counter and decide whether the binary zero or binary one was sent. Bits are shifted to the register and after minute synchronization pulse data are sent to the user application.

Figure 11 shows sample code of pulse decoder, which is used to determine synchronization pulses and received data. There is a possibility of receiving disturbing pulses in a shorter interval than one second, to be able to filter such pulses, the positive ones are stored into a buffer of length *PULSE_STORE_LEN* and the algorithm is waiting for the one in the time interval defined by the constant *TIME_LOW* and *TIME_HIGH*. When no pulse is received in this interval and up to the time *TIME_MIN_HIGH* it is determined as a minute pulse. The next steps like a time counter and its synchronization are more of the application layer and it is the business of target application.

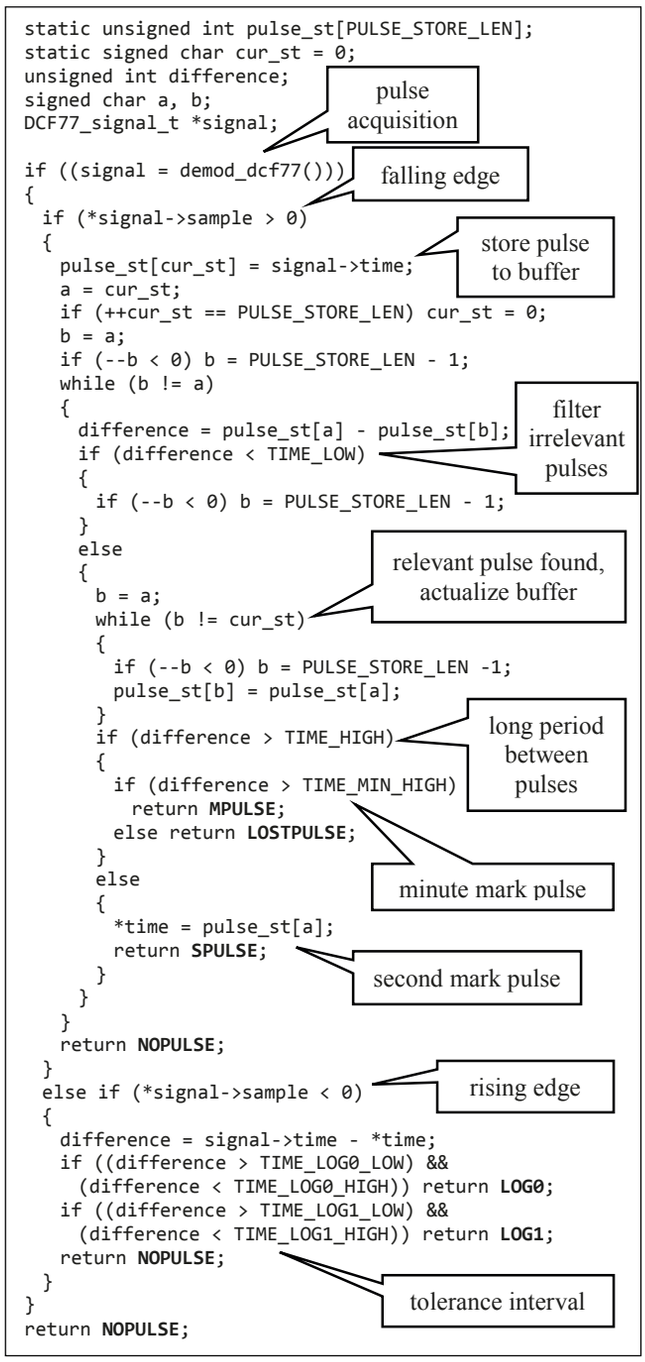

Figure 11. DCF77 pulse decoder code.

V. CONCLUSION

The DCF77 system was introduced. The software implementation is a modern technique of the receiver and for DCF77 seems to be very effective and applicable into modern consumer equipment.

Due to the character of DCF77 signal bandpass sampling technique is a good solution for AD conversion and downconversion all in one. The software receiver was intended to be implemented into ARM7 core LPC2103 microcontroller. Setting of the clocks and AD converter was defined and the following dealt with the whole chain implementation in GCC code. Concrete solutions of implementation of each block were proposed also with the discussion mainly of overflowing risks.

The software was already tested and is fully functional. The constants like an averager coefficients were evaluated experimentally and depends on the concrete hardware implementation, therefore these were not presented.

ACKNOWLEDGMENT

The research was performed in laboratories supported by the SIX project no. CZ.1.05/2.1.00/03.0072, the operation program Research and Development for Innovation and by the Wireless communication teams project no. EE2.3.20.0007.

REFERENCES

[1] HKW-Elektronik GmbH, Germany. “UE6005 Time-Code Receiver IC” (datasheet). 14 pages. [Online]. 2001, Rev. A10, Cited 2011-12-19. Available at: http://www.datasheetarchive.com/dl/Datasheets-316/627876.pdf

[2] TEMIC TELEFUNKEN microelectronic GmbH, Germany. “U4221B Radio Controlled Clock Receiver” (datasheet). 12 pages. [Online]. 1996, Rev. A1, Cited 2012-6-20. Available at: http://www.datasheetcatalog.org/datasheet/Temic/mXyzuryz.pdf

[3] Micro Analog Systems Oy, Finland. “MAS6180 AM Receiver IC” (datasheet). 14 pages. [Online]. 2011, Rev. DA6180B.002, Cited 2012-11-8. Available at: http://www.mas-oy.com/uploads/Data%20sheets/da6180B.pdf

[4] C-MAX Time Solutions GmbH, Germany. “CME8000 RC Receiver IC” (datasheet). 21 pages. [Online]. 2005, Rev. B.11, Cited 2012-11-8. Available at: http://datasheet.octopart.com/CME8000-DDTB-C-Max-datasheet-46588.pdf

[5] BAUCH, A., HETZEL, P., PIESTER, D. BAUCH, Time and Frequency Dissemination with DCF77: From 1959 to 2009 and beyond. “PTB-Mitteilungen: Amts- und Mitteilungsblatt der Physikalisch-Technischen Bundesanstalt”. 2009, No. 3. ISSN 0030-834X. Dostupné z: http://www.ptb.de/cms/fileadmin/internet/fachabteilungen/abteilung_4/4.4_zeit_und_frequenz/pdf/PTBM_50a_DCF77_engl.pdf

[6] VAUGHAN, Rodney G.; SCOTT, Neil L.; WHITE, D. Rod. “The Theory of Bandpass Sampling”. IEEE Transactions on Signal Processing [online]. Září 1991, 39, 9, [cit. 2011-08-02]. s. 1973-1984. Dostupný z WWW: <http://ieeexplore.ieee.org>. ISSN 1053-587X.

[7] OPPENHEIM, A. V., SCHAFER, R. W., BUCK, J. R. “Discrete-time signal processing”. 2nd ed. Upper Saddle River, NJ: Prentice Hall, 1999, 870 s. ISBN 01-375-4920-2.

[8] ZÁPLATA, F.; KASAL, M. “Software defined DCF77 receiver”. Radioengineering 2013 (sent to publish)