FPGA Implementation of Digital Modulation Recognition
I. A. Ziedan¹, M. I. Abdalla², O. M. Elkomy³

Abstract
This paper introduces a practical implementation of automatic modulation recognition (AMR) algorithm using Field Programmable Gate Array (FPGA). FPGA is an attractive choice for the implementation due to its advantages in performance, power consumption and configurability. Most of AMR researches are based on complex techniques that are not suitable for real time implementation, which requires an efficient use of the FPGA resources to reduce the number of gates. A new algorithm for AMR is introduced to detect the type of digital modulation presented in the incoming signal. This algorithm is carefully designed to meet the requirements of FPGA to obtain a single module that can be fitted in one chip. The modulation types considered are amplitude shift keying ASK, frequency shift keying FSK, and phase shift keying PSK, with a modulation degree up to 16-ary. The implementation is carried out using the SPARTAN-3 development kit from Xilinx[1]. The system can tolerate signal to noise ratio (S/N) down to 10 dB without errors.

KEYWORDS
FPGA, AMR, ASK, FSK, PSK, SPARTAN-3 Xilinx.

1 Prof., Faculty of Eng. Zagazig Univ.
2 Associate Prof., Faculty of Eng. Zagazig Univ.
3 Faculty of Computer Science and Informatics. Zagazig Univ.
1- Introduction

Automatic digital modulation recognition is a rapidly evolving area in communication system research where it covers many interesting applications in both civilian and military domains. These applications include signal monitoring, interference, intelligent modems, spectrum management, military threat detection and adaptive wireless communications[2].

Modulation recognition is an essential part in software radio[3,4], where the modulation scheme can be varied according to channel capacity, and a receiver with automatic modulation recognition is required to detect the modulation scheme in real time.

Many techniques for AMR have been published which are divided into three categories:

1- Decision theoretic approach
2- Pattern recognition approach
3- Combined approach

In decision theoretic approach the classification depends on signal envelope characteristics, zero-crossing, likelihood functions, and other statistical parameters like moments[5,6].

Pattern recognition approach is based on the principle of pattern recognition techniques which include signal preprocessing, features extraction, and finally the classifier.

Many researches used artificial neural network (ANN) as the classifier in modulation recognition for different types of digital modulation schemes[7,8,9,10].

A combined approach which uses more than one technique for modulation classification such as Fuzzy logic with statistical moment[11], hierarchical classification using cumulants[12], Discrete Fourier Transform (DFT) with phase histogram[13], joint phase lock detection and identification[14], fuzzy logic with genetic algorithm[15], and neural networks with higher order statistics[16].

Also there are some approaches which depend on the implementation method. For practical reasons considering complexity and hardware size, most of the previous algorithm can not be implemented for real time processing, so another simplified approaches are adopted to make the implementation feasible using hardware platforms like FPGA, Digital Signal Processors, and Microcontrollers.

This work is focused on the use of FPGA as the hardware platform for the implementation of AMR algorithm which can classify the major types of digital modulation (ASK, FSK, and PSK).

The choice of FPGA as the platform for the implementation is based on the fact that it has many advantages over other platforms. FPGA can be
used as a reconfigurable hardware to be adapted for various modulation schemes with the aid of simple microcontroller circuit and memory to hold the configuration data.

FPGA is based on hardwired logic circuits which provide a very high processing speed which is needed for communication signal processing. It also has the advantage of being software programmable using HDL (High Descriptive language) over discrete hardwired circuits which need complex design techniques.

2-Proposed Technique

The aim of the work is to present an algorithm of AMR for digital modulation schemes that can discriminate between 16-level QAM, 16-ary FSK, and 16-ary PSK.

Each type of the three signals is assumed to have the same carrier frequency. Other parameters like sampling frequency, carrier frequency, and synchronous carrier needed for signal processing are generated using analog front end processor.

All treated data are the samples of the incoming signal taken at a rate determined by the front end processor. These samples are constrained to 8-bit word length including the sign bit.

The block diagram of the proposed system is shown in Fig. 1. The signal is analyzed to its main parameters amplitude, phase, and frequency. These parameters are then analyzed using decision block to measure the variations in each parameter separately, then the system decides which type of modulation is present.

Only one of the three possible outputs will be active FSK, ASK or PSK, also the degree of modulation will be available as a binary code through data outputs $D_3D_2D_1D_0$ which represent a number from 1 to 16; for example if the modulation detected is PSK, and DATA= (0010), this means that it is a 2-ary PSK.

The following discussion explains the construction of each block in more details.

2.1 ASK Modulation

ASK modulation detection is based on observing the amplitude of the incoming signal and analyzing it to determine if there is any amplitude variation. If amplitude variation is observed then the next step is to decide the number of possible amplitudes the signal can have (degree of modulation).

Fig. 2 shows the block diagram of the proposed amplitude detection circuit. The signal is first squared and then averaged over one period to give an indication of signal amplitude as follows:
\[ y_1 = A^2 \left( \cos(\omega t + \theta) \right)^2 = \frac{A^2}{2} (1 + \cos(2\omega t + 2\theta)) \]  
(1)

\[ y_2 = \frac{A^2}{2} \]  
(2)

We get \( y_2 \) which is proportional to the amplitude of the incoming signal, and the next step is to determine if this amplitude is almost constant (no modulation present) or varies between different values.

Another block is added to perform amplitude analysis as shown in Fig. 3. The 16-value discriminator is used to analyze the amplitude variation to determine the presence of modulation and its degree. The amplitude values are averaged and stored in an 8-bit register. Only the 4 high order bits are used to measure amplitude variation, and this gives a 16 possible amplitude values.

The 4 to 16 decoder will activate one of its outputs according to the value of the incoming amplitude. This active signal is used to increment a counter associated with this amplitude to count the number of occurrences of each amplitude value over 96 cycles. Finally after 96 cycles the 16 to 5 encoder gives a number from 1 to 16 indicating the degree of modulation if present. To clarify this function, consider that the modulation has a degree of 4 then signal amplitude will vary between 4 different values (\( A_1 \), \( A_2 \), \( A_3 \), \( A_4 \)).

Now if the current amplitude is \( A_i \) (\( i=1,2,3,4 \)) then counter will be incremented and finally after 96 cycles we get only 4 non-empty counters indicating that a modulation is present and its degree is 4. If only we get one non-empty counter, this indicates that there is one signal amplitude (no amplitude variations), and the 16 to 5 encoder will indicate that no modulation is present.

The 3rd bit of each counter is used to indicate if the counter is full or empty. The design of the 16 to 5 encoder is simplified using a 16 bit adder, where the number of full counters is summed to give an indication of the modulation degree.

### 2.2 FSK Modulation

Fig. 4 shows the details of the frequency detector block, where a counter with fixed clock frequency is used to measure the period of each signal cycle.

A zero cross detector is used to trigger the counting operation by observing the signal variation from negative value to positive value, and this can be simplified by using the sign bit of the input signal (1 negative value- 0 positive value) as the counter enable signal. After each cycle the counter value is latched into a register to be processed further to determine if modulation is present and the degree of modulation, and this can be done using the same block (16-value discriminator) used in ASK detection.
2.3 PSK Modulation

The proposed system for m-ary PSK modulation detection is based on measuring the phase difference \( \theta \) by using synchronous demodulator as shown in Fig. 5 where:

\[
y_1 = A^2 \cos(\omega_c t) \cos(\omega_c t + \theta)
\]

\[
y_1 = \frac{A^2}{2} (\cos(2\omega_c t + \theta) + \cos(\theta))
\]

The first term will be filtered out and we get

\[
y_o = \frac{A^2}{2} \cos(\theta)
\]

The output now is proportional to the phase difference \( \theta \) and can be used to measure the angle of the incoming signal, but this angle is restricted to values between 0 to 180 degree where we can't distinguish a phase difference of 90 and -90 degree.

So we must add another parameter to extend the range of measurements to any value between 0 and 360 degree. Another branch is added as shown in Fig. 6

We have two outputs:

\[
y_{o1} = \frac{A^2}{2} \cos(\theta)
\]

\[
y_{o2} = \frac{A^2}{2} \sin(\theta)
\]

The first output can be used to measure the value of the phase difference and the other output is used to determine the polarity of the phase difference and hence we can measure a phase difference between -180 and up to 180 degree.

The implementation of the proposed m-ary PSK modulation detection using FPGA module must be modified to satisfy the design consideration and capabilities of FPGA.

The main functions of the proposed technique are:

1- Multiplication
2- Low pass filtering

2.3.1 Multiplication

The proposed technique needs to multiply two sampled signals quantized by 8 bits each. This multiplication will consume most of the FPGA cells if it is implemented by direct hardware multiplier, where a multiplication of two operands of 8 bits each require an array of \( 8 \times 8 \) full adders which will consume large amount of FPGA cells.

So another technique is used to perform multiplication using memory lookup tables, where the values of multiplication can be stored in a memory for each expected combination of input values. For \( 8 \times 8 \) multiplication we will need \( 2^{16} \) memory location.
The multiplication process in the proposed algorithm must be modified to make the implementation using FPGA module more convenient. Fig. 7 shows how to perform signal multiplication using indirect method that will lead to more efficient use of FPGA.

\[
y_1 = A \left( \cos(\omega_t t) + \cos(\omega_t t + \theta) \right)
\]

\[
y_1^2 = A^2 \left( \cos(\omega_t t) + \cos(\omega_t t + \theta) \right)^2
= A^2 \left( \cos(\omega_t t) \right)^2 + \left( \cos(\omega_t t + \theta) \right)^2
+ 2 \cos(\omega_t t) \cos(\omega_t t + \theta)
\]

\[
y_1 = A^2 \left( \frac{1}{2} (1 + \cos(2\omega_t t)) + \frac{1}{2} (1 + \cos(2\omega_t t + 2\theta)) + \cos(\theta) + \cos(2\omega_t t + \theta) \right)
\]

After the LPF we get,

\[
y_{o1} = A^2 \left( 1 + \cos(\theta) \right)
\]

Similarly the orthogonal branch shown in Fig. 8 will produce,

\[
y_{o2} = A^2 \left( 1 + \sin(\theta) \right)
\]

The squaring process can be implemented in FPGA using lookup table with 8-bits input and a normalized 8 bits output will require 256 bytes of memory space.

The same table can be duplicated to calculate the output of the second branch.

### 2.3.2 Low pass filtering

The design of LPF using DSP techniques like finite impulse response filter (FIR) will consume large amount of the FPGA chip, so we simplify the design of the filter by using the average process over one period of the input signal that will be suitable for FPGA implementation.

Fig. 9 shows the block circuit of average filter used to reject the high frequency terms and the noise from the processed signal.

### 3 Results and Discussion

The proposed system was built using Xilinx Spartan-3 platform FPGA development kit which contains 200,000 gates Platform FPGA XC3S200-4FT256C Chip. The development kit contains 1M-byte of fast asynchronous SRAM which is used to store the lookup tables needed for implementation.

Table 1 shows FPGA usage for each type of modulation recognition used in the implementation. The FPGA usage is measured in units of CLB (Configurable Logic Block) which is the smallest functional block.

Also the maximum clock frequency that the implementation can handle is indicated in Table 1. This frequency
indicates how fast the system for real time applications.

The proposed system needs 979 CLBs to be implemented which can be fitted in one large sized FPGA like Spartan-3. The clock frequency is limited to 77.3 MHz to accommodate the PSK detection requirements and is adequate for real time system.

Also since the entire AMR algorithm is fitted in one chip FPGA, there will be no need to reconfigure the chip for each modulation type like other system[17].

The system was tested using a test setup consisting of microcontroller based signal generator and a lookup memory for modulation selection. The test signal is sampled at 20 MHz sampling rate. The digital data is taken from a predefined file stored in the ROM of the microcontroller with data rate of 512 kbps.

The proposed squaring process reduces the signal to noise ratio at the output of the multiplier and this affect the maximum degree of modulation that the system can handle. A modification is added to the system to reduce the noise by increasing the resolution of the multiplier using 12 bit lookup tables and filters. Increasing the resolution to reduce system noise enables the system to handle modulation degree up to 16 levels without errors, but a remarkable increase in CLB usage is observed. Fortunately the total CLB usage is within the limit of the SPARTAN-3 FPGA chip (1020 CLBs).

Further increase in the degree of modulation will need a larger chip or the use of two chips to enable higher resolution.

The proposed system was tested under noisy environment with SNR of 10 dB and the success rate is 100% (without errors). This result represents a performance advantage over previous researches in AMR [9,15,16,17].

4 Conclusion

The proposed system and FPGA implementation of AMR works successfully for ASK, FSK, and PSK. Modulation degree up to 16-ary was achieved without error. Only one FPGA chip is used for implementation without the need for reconfiguration process which increases the detection time. The proposed system proves that FPGA can be a suitable platform for AMR implementation for real time application and with the increasing progress in the development of FPGA architecture more modulation scheme can be fitted in a single chip. Also the implementation of demodulators can be easily achieved with simple modification of the proposed hardware. So a complete module of demodulator with AMR can be implemented using FPGA hardware suitable for many applications such as software radio.
References


Figure 1 Block Diagram of Modulation Detection
Figure 2 Block Diagram of ASK Detection

Figure 3. Block Diagram of 16 value discriminator

Figure 4. Block Diagram of FSK Detection
Figure 5. Block Diagram of simplified PSK Detection

![Block Diagram of simplified PSK Detection](image)

Figure 6. Block Diagram of proposed PSK Detection

![Block Diagram of proposed PSK Detection](image)

Figure 7. Block Diagram of multiplication process

![Block Diagram of multiplication process](image)
Figure 8 Block Diagram of multiplication process in the orthogonal branch

Figure 9 Block Diagram of the average filter

Table 1 Performance measure (CLB usage and Maximum usable frequency)

<table>
<thead>
<tr>
<th>Modulation</th>
<th>CLBs / Max CLBs</th>
<th>Max Frequency Mhz</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASK</td>
<td>301/1024</td>
<td>81.7</td>
</tr>
<tr>
<td>FSK</td>
<td>215/1024</td>
<td>95</td>
</tr>
<tr>
<td>PSK</td>
<td>463/1024</td>
<td>77.3</td>
</tr>
</tbody>
</table>