© 2018 IJSRST | Volume 4 | Issue 5 | Print ISSN: 2395-6011 | Online ISSN: 2395-602X
Themed Section: Science and Technology

Optimized ARM-based implementation of DAB Time and Frequency Synchronization

S. Anupriyadharsini[1], R. Sarah Blessie[2], B. Jeba Sangeetha[3], Dr. A. Banumathi[4]
[1] M.E-Communication Systems, Thiagarajar college of Engineering ,Madurai, Tamil Nadu, India
[2] System Engineer, Jasmin Infotech Pvt.Ltd,Chennai, Tamil Nadu, India
[3] Project Leader, Jasmin Infotech Pvt.Ltd, Chennai, Tamil Nadu, India
[3]Associate Professor, Thiagarajar College of Engineering, Madurai, Tamil Nadu, India

ABSTRACT

All digital communication systems require proper synchronization for decoding of the received signal in order to obtain the original information transmitted. In Digital Audio Broadcasting (DAB), Synchronization plays an important role by locating each frame precisely. Digital Audio Broadcasting system explores the Null symbol and the phase reference symbol for synchronization purpose. Software Defined Radio-DAB is a system where components that have been traditionally implemented in hardware (example: Filters, modulators/ demodulators) are implemented by means of software on a computer or embedded system. Traditional hardware based radio devices can only be modified through physical invention. This results in higher production costs and minimal flexibility in supporting multiple waveforms standards. Whereas Software Defined Radio technology provides an efficient and comparatively inexpensive solution allowing multiple mode, multi-band and multi-functional wireless devices that can be enhanced using software upgrades. However, a pure software solution (Software Defined Radio) consumes more memory than a hardware solution. Existing techniques are still challenging in optimizing the memory requirement for Software Defined Radio to achieve overall cost effectiveness and flexibility. This work aims to produce an efficient algorithm for reducing the memory requirement for SDR DAB.

Keywords: DAB(Digital Audio Broadcasting), Synchronization, Float to fixed, PRS (Phase Reference Symbol), Partial symbol based

I. INTRODUCTION

The new digital radio system DAB (Digital Audio Broadcasting), is a very innovative and universal multimedia broadcast system which will replace the existing AM and FM audio broadcast services in future. DAB is able to transmit programme-associated data and a multiplex of other data services (e.g. travel and traffic information, pictures, etc.). DAB System can be operated at any frequency from 30 MHz to 3GHz for mobile reception. It is adopted by the European Telecommunications Standards Institute(ETSI).

A. Main System Features

The DAB transmission signal carries a multiplex of several digital services (audio and data) simultaneously. Its overall bandwidth is 1.536 MHz, and a useful bit-rate capacity of approximately 1.5 Mbit/s in a complete "ensemble".

• Each service is independently error protected with a coding overhead ranging from 25% to 300% (25% to 200% for sound), the amount of which depends on the requirements of the broadcasters (transmitter coverage, reception quality)[3].

• A specific part of the multiplex contains information on how the multiplex is configured, so that the receiver can decode the signal correctly. It may also carry information about the services and the links between different services.

B. DAB Transmission system

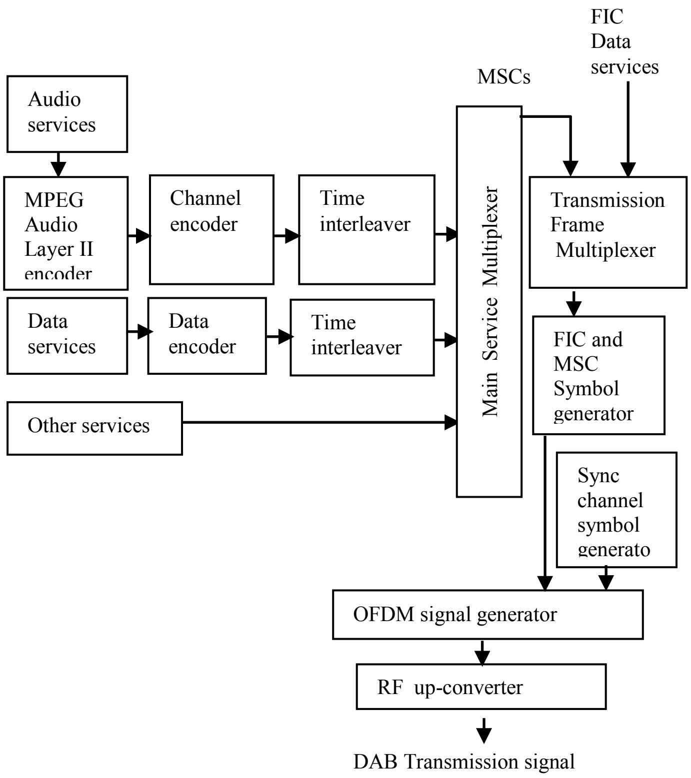


Figure 1. DAB Transmission system

Figure 1 represents the transmission system of DAB.

- Audio data as well as other data is individually encoded with channel encoders and then error protected and time interleaved[2].
- The services are multiplexed in the Main Service Channel (MSC).
- The multiplexer output is then combined with Multiplex Control and Service Information(SI) in the Fast Information Channel (FIC) to form the transmission frames in the Transmission Multiplexer. The primary function of FIC which is made of Fast Information Blocks(FIB) is to carry control information necessary to interpret the configuration of the MSC. The essential part of the control information is the Multiplex Configuration Information (MCI)[5].
- DAB employs COFDM modulation which combines the multi-carrier modulation technique-OFDM (Orthogonal Frequency Division Multiplexing) with convolutional channel coding in such a way that the system can exploit both time and frequency diversity[4].
- The signal is then finally transposed to the appropriate radio frequency band, amplified and transmitted.

DAB system has four transmission modes of operation namely mode-I, mode-II, mode- III, and mode-IV, each having its own set of parameters. The use of these transmission modes depends on the configuration of the network and operating frequencies. This makes the DAB system operate over a wide range of frequencies from 30 MHz to 3 GHz.

DAB transmission signal s(t) is given by

$$\boldsymbol{s(t) = Re\{e^{j2\pi f_c t} \sum_{m=-\infty}^{\infty} \sum_{l=0}^{L} \sum_{k=-\frac{K}{2}}^{\frac{K}{2}} z_{m.l.k} \times g_{k,l}(t - mT_F - T_{NULL} - (l-1)T_S)\}} \quad (1)$$

Where,

$$g_{k,l}(t) = \begin{cases} 0, & for\ l = 0 \\ e^{\frac{2j\pi k(t-\Delta)}{T_U}}.Rect\left(\frac{t}{T_S}\right), & for\ l = 1,2 \ldots L \end{cases} \quad (2)$$

$$T_s = T_U + \Delta$$

Where,

L=number of OFDM symbols per transmission frame
K=number of transmitted carriers
T_F = transmission frame duration
T_{NULL}=NULL symbol duration

T_S=duration of OFDM symbols of indices l=1,2,3…L
T_U=inverse of carrier spacing
Δ=duration of time interval (guard interval)
f_c= central frequency of the signal

$z_{m.l.k}$= complex D-QPSK symbol associated with carrier k of OFDM symbol l during transmission frame m.

The DAB transmission system combines three channels as shown in Fig.2

Main Service Channel (MSC): It is used to carry audio services and data service components. The MSC is a time interleaved data channel divided into a number of sub-channels that are individually convolutionally coded, with equal or unequal error protection schemes. Each sub-channel may carry one or more service components.

Fast Information Channel (FIC): It is used to send the Multiplex Configuration Information (MCI), Service Information and data services. FIC is a non-time-interleaved data channel with fixed equal error protection.

Synchronization channel: It is used internally within the transmission system for basic demodulator functions, such as frame synchronization, automatic frequency control, channel state estimation, and transmitter identification.

Synchronization Channel	Fast Information Channel(FIC)	Main Service Channel(MSC)

Figure 2. Transmission frame

II. SYNCHRONIZATION

Synchronization is challenging and plays a major role in digital communication system. There are two problems in designing the receiver. One problem is the unknown symbol arrival time and second problem is the mismatch of the oscillator in the transmitter and the receiver. These problems should be eliminated. Hence the receiver needs to know about the carrier phase, carrier frequency offset, symbol timing and frame timing for synchronization. Synchronization is the key part of OFDM receiver. A method to estimate Time Synchronization in DAB receiver is based on analysis of phase reference symbol(PRS). It is used to locate precisely each DAB frame. DAB system explores the Null symbol and the phase reference symbol for synchronization purpose.Figure 3 represents the synchronization channel. Since DAB uses OFDM digital transmission technique, the synchronization block plays an important role in determining bit error rate performance of DAB system in different channels.

NULL	PRS	FIC	MSC

Sync channel

Figure 3. Synchronization Channel

A. Null symbol

The first OFDM symbol of transmission frame is the null symbol. During the time interval [0,T_{NULL}], the main signal s(t) is equal to 0. During null symbol period no information is transmitted.

B. Phase reference symbol

The next OFDM symbol of the transmission frame is the phase reference symbol. It constitutes the reference for the differential modulation for the next OFDM symbol. The phase reference symbol is defined by the values of z_k.

$$z_k = \begin{cases} e^{j\varphi_k} & for\ -\frac{K}{2} \leq k < 0\ and\ 0 < k \leq \frac{K}{2} \\ 0 & for\ k = 0 \end{cases} \quad (3)$$

The values of φ_k can be obtained from the below formula

$$\varphi_k = \frac{\pi}{2}(h_{i,k-k'} + n) \quad (4)$$

The indices i, k' and the parameter n are specified as functions of the carrier index k.

III. PROPOSED METHODOLOGY

In order to reduce the memory requirement of Time and Frequency Synchronization module in SDR-DAB, the following methods were adopted(Fig. 4)

- Float to fixed implementation
- Partial Symbol based implementation

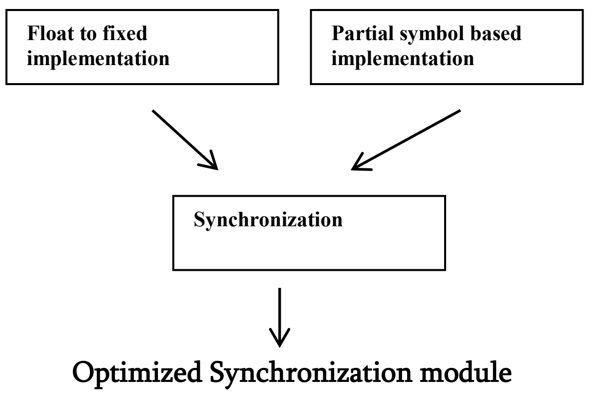


Fig.4 Proposed methodology

A. Float to fixed implementation

Computers use binary digits to represent numbers and other data. The computer memory is organized into strings of bits called words. Decimal numbers are first converted into binary equivalents and then they are represented in either integer or floating point form.

Most modern computers have hardware support for floating point numbers. However the use of floating point representation is not necessarily the only way to represent fractional numbers The fixed point data type is used widely in digital signal processing(DSP) where performance is sometimes more important than precision. Fixed point arithmetic operation is much faster than a floating point arithmetic operation

B. Floating point representation

Floating point refers to the fact that a number's radix point (decimal point) can float that is it can be placed anywhere relative to the significant digits of the respective number. This position is indicated as the exponent component. Since 1990s the most common representation of floating point is defined by IEEE 754 standard.(Fig. 5).

S	E	M

S - Sign 1(-ve),0(+ve)}

E - Exponent---more exponent bits ⟹ greater range

M - Mantissa---more significand bits ⟹ greater accuracy

Figure 5. Floating point representation

The precision to which the numbers can be represented is determined by the length of the significand. The radix point position is assumed always to be somewhere within the significand just after or just before the most significant digit or to the right of the rightmost (least significant) digit (signed integer exponent also known as the characteristic or scale which modifies the magnitude of the number).

Representation of Floating point numbers in Single Precision IEEE 754 Standard

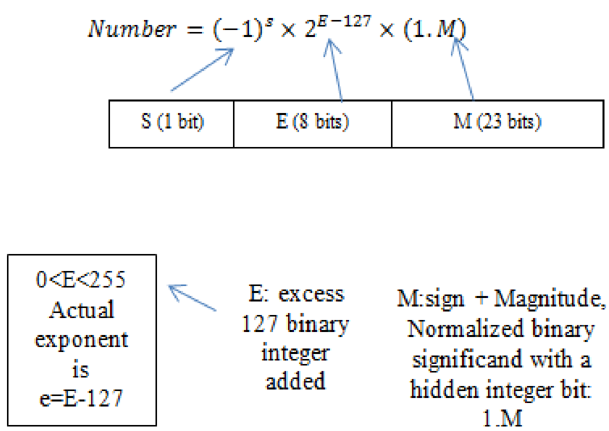


Figure 6. Single precision representation

Representation of Floating point numbers in Double Precision IEEE 754 Standard

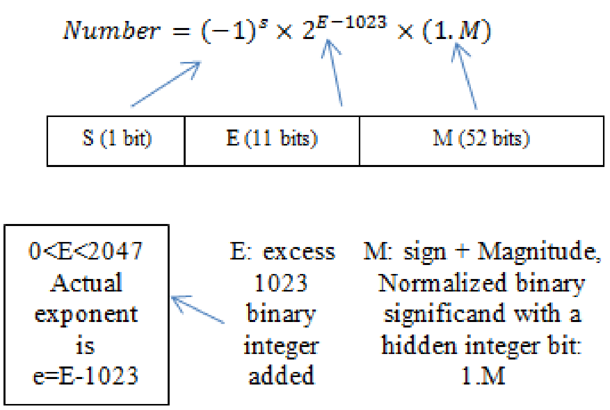


Figure 7. Double precision representation

The IEEE Floating Point Standard defines rounding rules for choosing the closest floating point when a rounding error occurs:

RN - Round to Nearest.

RZ - Round toward Zero. Same as truncation in sign-magnitude.

RP - Round toward Positive infinity.

RM -Round toward minus infinity. Same as truncation in integer 2's complement arithmetic.

RN - is generally preferred and introduces less systematic error than the other rules.

C. Fixed point number representation

To improve mathematical throughput and to increase the execution rate, calculations can be performed using twos complement signed fixed point representations.

Fixed point notation is also a representation of the fractional number as it is stored in memory. In fixed point representation the number is stored as a signed integer in two's complement format. Fixed point representation requires the user to create a virtual decimal place in between two-bit locations for a given length of data.

In order to specify the number of bits needed to represent the integer and fractional parts of the number a notation called the Q format is used

Qm.n

m bits for integer portion

n bits for fractional portion

Total number of bits N=m+n+1,1 for signed numbers

Fixed-point numbers are represented by the following characteristics:

1. The word length in bits
2. The position of the binary point
3. Whether it is signed or unsigned

Example:

8	4	2	1		0.5	0.25	0.125	0.0625
0	1	1	0	.	1	0	0	0

The fixed-point values are scaled by means of the position of the binary point. The signed numbers are used to represent both positive and negative numbers. Whereas, the unsigned numbers are used for positive numeric data only. The position of the binary point is represented by numbers m and n, where m is the number of bits on the left side of the binary point, called the integer word length, and n is the number of bits on the right side of the binary point, which is called the fraction word length.

Unsigned Fixed-Point Numbers

Unsigned fixed-point numbers are used to denote numbers greater than or equal to zero. The value of an unsigned N-bit binary number, x, with an integer word length a and fraction word length b is given by

$$x = \frac{1}{2^b}\sum_{n=0}^{N-1} 2^n x_n \qquad (5)$$

Where x_n is is bit n of x.

Therefore, the range of an unsigned number is between 0 and $\frac{2^N-1}{2^b} = 2^a - 2^b$. a and b can have negative values to represent very small or very large values.

Signed Fixed-Point Numbers

Signed fixed-point numbers are used to denote both positive and negative numbers. The first bit on the left side of the number is used for the sign of the number. Therefore, for an N-bit number with a fraction word length of b, the integer word length is equal to $a = N - b - 1$.The value of a given N bit signed number, x, is given by

$$x = \frac{1}{2^b}[-2^{N-1}x_{N-1}+\sum_{n=0}^{N-2} 2^n x_n] \quad (6)$$

Where $x_n$is bit n of x. The range of an N bit signed number is between -2^{N-1-b} and $2^{N-1-b} - 2^{-b}$.

D. Considerations while setting Q format

The variable to be converted is explored for

- **Range** of the variable value
- Absolute **Maximum / Minimum** values of variable
- **Resolution** of variable: smallest non zero number
- **Accuracy**: Maximum error for a representation. Accuracy required highly depends on the arithmetic involved. Accuracy can be calculated from **tolerable %error** in representation.
- **Precision of Fixed-Point Number**: Precision of a fixed-point number is equal to the word length. For example, precision of a 16-bit signed number is 16 bits.
- **Resolution of a Fixed-Point Number:** Resolution depends on the smallest non-zero magnitude that can be represented.

E. Floating-point to Fixed-point conversion

To convert a floating-point value to the corresponding fixed-point value the following steps are involved:
Consider a floating-point variable, a:

Step 1: Calculate b=a*2^F , where F is the fractional length of the variable. b is represented in decimal.

Step 2: Round the value of b to the nearest integer value. For example:

Round (3.66) = 4

Round (-1.6) = -2

Step 3: Convert b from decimal to binary representation and name the new variable c.

The conversion needs to be done in all the arithmetic operations involved.

Fixed-point Arithmetic

The arithmetic operations are addition, subtraction, multiplication, and division among which addition and multiplication play an important role. Since subtraction is the inverse of addition and division is equivalent to multiplication by the multiplicative inverse,the shifting operation is used to perform addition and multiplication. A fixed-point number with the integer word length of a and fractional word length of b is taken as x(a; b).

Shifting

Shifting operation is equivalent to multiplication or division by a power of two. Shifting is also used to displace the position of the binary point. Displacement of the binary point is performed in addition and multiplication operations. A shift can be performed either to the right or to the left. The number of shifts must be a positive integer number. Shifts to the right and to the left are indicated by >> and <<, respectively. If n is the number of shifts, we have

x (a; b) >> n = x (a - n; b + n)

x (a; b) << n = x (a + n; b - n):

Addition

In order to add two fixed-point numbers, they must be scaled to the same format, that is, x (a; b)+y (c; d) has a correct result on the condition that a = c and b = d. Therefore, first they must be shifted to have the same binary point position and then added to each other. The result of the addition is a fixed-point number with a length of N + 1 bits, where N is the number of bit in each number.

Multiplication

For multiplication, both numbers must be either signed or unsigned. In unsigned multiplication, the result has a number of bits equal to the sum of the numbers of the bits of the multiplicatives.

x (a1; b1) + y (a2; b2) = z (a1 + a2; b1 + b2)

For signed multiplication, the result is as follows:

x (a1; b1) + y (a2; b2) = z (a1 + a2 + 1; b1 + b2)

F. Partial symbol based implementation

Current implementation of synchronization module process one frame of data (196608 samples (76 OFDM+NULL symbol) for mode I.

(i.e) 2656(Null)+76*(2048(Useful)+504(Guard)) =196608

Partial Symbol based method allows us to process symbol by symbol in the synchronization module hence the storage size will be reduced.In this work the steps followed for partial symbol based technique are given in Fig. 8.

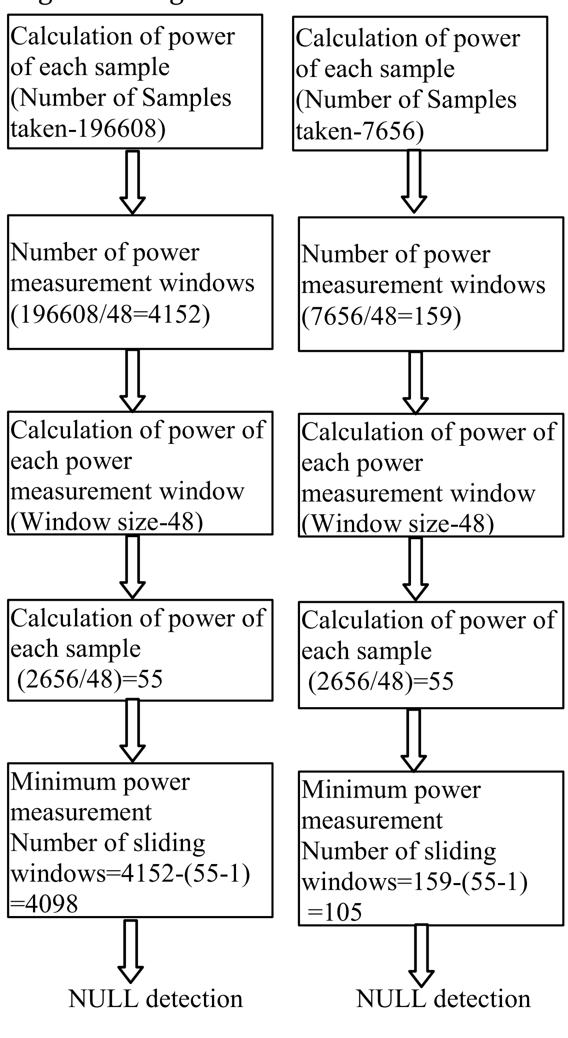


Figure 8. Partial algorithm for NULL detection

The above steps were followed while finding Null position. Here 3 symbols were considered, hence 3*2552=7656.Therefore samples are reduced from 196608 to 7656.

IV. RESULTS AND DISCUSSIONS

Input=196608 samples(Mode-1)

The samples include 76 ofdm symbols and one Null symbol

The input data is pointed to by a buffer. It is an array of type **float** containing the sampled data in format I(imaginary), Q(real) , I , Q,... (**Complex** variables)

Therefore the size of the input variable is=196608*2(complex)*4(float)

A. DAB mode and parameters involved

Null window=48

MODE-1

Input = 196608 samples (76 OFDM+NULL symbol)

Size=196608*2*4

Useful data =2048

Guard band= 504

Number of symbols per frame= 76

Sampling frequency=1.0 kHz

Search window= 2600

Symbol Size = Guard Size +Useful Size;

=504+2058

=2552

Therefore input= (76*2552) +size of one null symbol(2656)

=196608

MODE-2

Input= 49152(76 OFDM+NULL symbol)

Size=49152*2*4

Useful data= 512

Guard band= 126

Number of symbols per frame= 76

Sampling frequency= 4.0 kHz

Search window= 600

Symbol Size = Guard Size +Useful Size;

=126+512

=638

MODE-3

Input= 49152 samples (153 ofdm symbol+1 Null symbol);

Useful data= 256

Guard band= 63

Number of symbols per frame= 153

Sampling frequency= 8.0 kHz

Search window= 300

Symbol Size = Guard Size +Useful Size;

=63+256

=319

MODE-4

Input=98304samples (76 ofdm symbol+1 Null symbol);;

Useful data= 1024

Guard band= 252

Number of symbols per frame= 76

Sampling frequency= 2.0 kHz

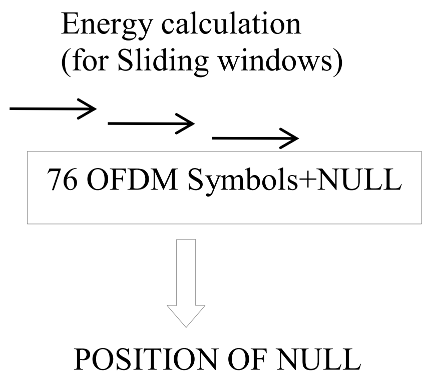
Search window= 1300

Symbol Size = Guard Size +Useful Size;

=252+102=1276

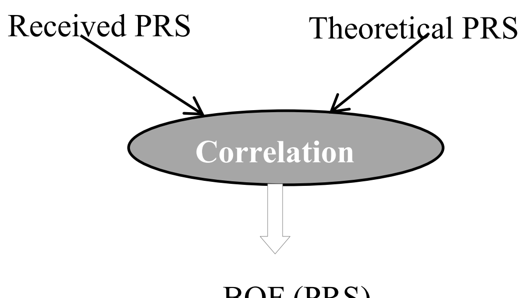
B. Synchronization

Finding Null



First null offset was observed to be at ---127607 (Fig.9)

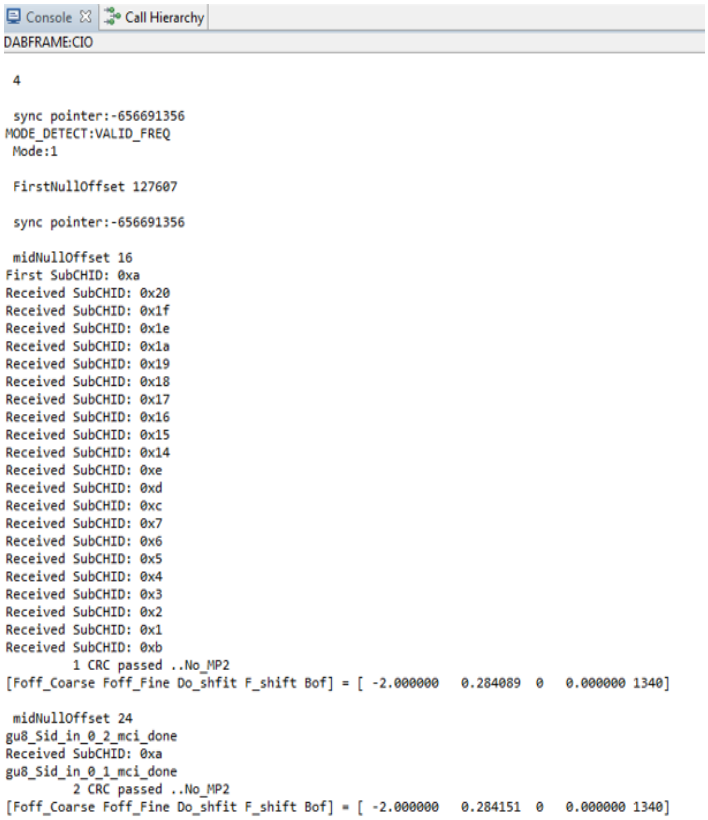
To find the beginning of frame



Beginningof frame was observed at,approximately--1340(Figure 9)

Coarse frequency offset estimation

Then by computing cross-correlation the Coarse frequency offset was observed to be –2 KHz (Fig.9)


```
Console  Call Hierarchy
DABFRAME:CIO

 4

 sync pointer:-656691356
MODE_DETECT:VALID_FREQ
 Mode:1

 FirstNullOffset 127607

 sync pointer:-656691356

 midNullOffset 16
First SubCHID: 0xa
Received SubCHID: 0x20
Received SubCHID: 0x1f
Received SubCHID: 0x1e
Received SubCHID: 0x1a
Received SubCHID: 0x19
Received SubCHID: 0x18
Received SubCHID: 0x17
Received SubCHID: 0x16
Received SubCHID: 0x15
Received SubCHID: 0x14
Received SubCHID: 0xe
Received SubCHID: 0xd
Received SubCHID: 0xc
Received SubCHID: 0x7
Received SubCHID: 0x6
Received SubCHID: 0x5
Received SubCHID: 0x4
Received SubCHID: 0x3
Received SubCHID: 0x2
Received SubCHID: 0x1
Received SubCHID: 0xb
        1 CRC passed ..No_MP2
[Foff_Coarse Foff_Fine Do_shfit F_shift Bof] = [ -2.000000   0.284089  0   0.000000 1340]

 midNullOffset 24
gu8_Sid_in_0_2_mci_done
Received SubCHID: 0xa
gu8_Sid_in_0_1_mci_done
        2 CRC passed ..No_MP2
[Foff_Coarse Foff_Fine Do_shfit F_shift Bof] = [ -2.000000   0.284151  0   0.000000 1340]
```

Figure 9. Null and Frequency offsets

Fine frequency offset estimation

Fine frequency offset was observed to be approximately 0.28 KHz(Figure 9)

C. Float to Fixed Implementation

In this work Floating point variables and the respective operations were converted to Q 15.0 format. By implementing algorithms using fixed point arithmetic (integer) a significant reduction in memory was observed.

D.Partial Implementation

Instead of considering 196608 samples 3 ofdm symbols

were considered one after the other.
(i.e) 3*useful data
=3*(2048+504)
=3*2552
=7656

The following steps describes the partial symbol based algorithm for Finding null.

Number of power measurement windows per read

Number of Power Windows = floor (Number of SamplesPerRead / null Window);

Number of Power Windows =7656/48=159;

Number of power measurement windows in one Null symbol

Windows In Null = floor (samples In Null / null Window);

Windows In Null =2656/48=55;

The sliding window size in number of complex samples

Sliding Window Size = Windows In Null * null Window;

Sliding Window Size =55*48=2640;

Number of sliding windows

Number of Sliding Windows = Number of Power Windows – (Windows In Null - 1);

Number of Sliding Windows=159-(55-1)=105;

Power of each sample was calculated

Power of each sliding window size was calculated

Minimum power was found (should be nearly equal to zero)

First Sample = Minimum index *48;

Last Sample = First Sample + Sliding Window Size - 1;

Middle Null = (First Sample + Last Sample)/2;

Table 1. Available memory

Module Name	**Data Memory (kB)**
Sync Module	4150.671875

Table 2. Reduced memory

Module Name	**Data Memory (kB)**
Sync Module	1559.2275

Memory after optimization =**2591.444375 Kb**

E. Observed BER values after optimization

```
        33 CRC passed
[Foff_Coarse Foff_Fine Do_shfit F_shift Bof] = [ -2.000000   0.285476  0   0.000000 1337]

 midNullOffset -292

BER_Val        : 0.000000

BER_Val        : 0.000000

BER_Val        : 0.000000

BER_Val        : 0.000000
        34 CRC passed
[Foff_Coarse Foff_Fine Do_shfit F_shift Bof] = [ -2.000000   0.285180  0   0.000000 1337]

 midNullOffset -192

BER_Val        : 0.000000

BER_Val        : 0.015092

 BER_Value:     1.509202 e - -2

BER_Val        : 0.000000

BER_Val        : 0.000000
        35 CRC passed
[Foff_Coarse Foff_Fine Do_shfit F_shift Bof] = [ -2.000000   0.286297  0   0.000000 1337]

 midNullOffset -112
```

The BER values were same when compared to previous BER values(before optimization) (Figure 10)

Then the same process was repeated using J6 Vayu EVM(Two ARM core A15) and MCPS was observed to be the same.

V. CONCLUSION

Efficient algorithms were designed for the optimization of synchronization module in SDR-DAB. Float to fixed point conversion technique and Partial symbol based technique were used and a memory of 1559 kB was reduced. In future,these optimization techniques can be applied to DAB as a whole.

VI. REFERENCES

[1]. A J Bower, "DIGITAL RADIO--The Eureka 147 DAB System," Electronic Engineering BBC, April 1998.
[2]. Ashish Agarwal, "DAB System: Transmitter, Receiver and SVM classifier," National Institute of Technology, Master's thesis, Rourkela,2011.
[3]. Baliwan Preeti, "Receiver Synchronization For DAB With Transmission Mode III," International Journal Of Engineering And Computer Science, Volume 3 Issue 2 February , 2014 Page No. 3830-3836.
[4]. ETSI TR 101 496-3 , "Digital Audio Broadcasting (DAB); Guidelines and rules for implementation and operation; Part 3: Broadcast network," V1.1.2 (2001-05).
[5]. ETSI TR 300 401,"Radio Broadcasting Systems ;Digital Audio Broadcasting(DAB)to mobile ,portable and fixed receivers.," V1.4.1 (2006-01).
[6]. Petro Pesha Ernest, "DAB implementation in SDR," University of Stellenbosch, Master's thesis ,December 2005.