

**RAMAIAH INSTITUTE OF TECHNOLOGY**

# (Autonomous Institute Affiliated to VTU), Bangalore

# Dept. of Electronics and Communication Engineering

**MINI PROJECT**

**REPORT ON**

**Generation of DSBSC waveform using LPC2148**

|  |  |
| --- | --- |
|  | **Report Submitted by** |
|  | Askanda Mahajan 1MS21EC030Jayshree V J 1MS21EC134Aditi Kulkarni 1MS21EC137 |

1. **Introduction**

Double Sideband Suppressed Carrier (DSB-SC) modulation is an efficient modulation technique where only the sidebands carry the information, and the carrier is suppressed. This approach conserves bandwidth and power, making it suitable for various communication applications. This report outlines the process of generating a DSB-SC signal using the LPC2148 microcontroller, from theoretical principles to practical implementation and testing.

1. **Objectives**

* To understand the principles and applications of DSB-SC modulation.
* To implement a DSB-SC signal generation algorithm using the LPC2148 microcontroller.
* To configure the DAC for analog signal output.
* To create and test the modulated signal.
* To outline the steps to create a hex file using the IAR Embedded Workbench IDE.

**3. DSB-SC Working**

DSB-SC modulation involves modulating a message signal with a carrier signal such that the carrier is suppressed and only the sidebands are transmitted. The mathematical representation of a DSB-SC signal is

s(t) = m(t) cos(omega\_c t)

where m(t) is the message signal and cos(omega\_c t) ) is the carrier signal.

By suppressing the carrier, the method achieves greater power efficiency and better bandwidth utilization

**4. Code Explanation**

The provided code generates a DSB-SC signal using the LPC2148 microcontroller's DAC.

**Code**

#include "lpc214x.h" // LPC2148 specific definitions

#include <stdint.h> // Standard integer types

#include <math.h> // Math functions

#define PI 3.14159265358979323846

#define SAMPLING\_RATE 1000 // Sampling rate in Hz

#define CARRIER\_FREQ 100 // Carrier frequency in Hz

#define MESSAGE\_FREQ 10 // Message frequency in Hz

void delay\_us(uint32\_t us){

uint32\_t i;

for(i = 0; i < us \* 15; i++); // Rough estimate for 1 microsecond delay

}

int main(){

uint16\_t sample;

float carrier, message, dsbsc;

float carrier\_phase = 0, message\_phase = 0;

float carrier\_step = 2 \* PI \* CARRIER\_FREQ / SAMPLING\_RATE;

float message\_step = 2 \* PI \* MESSAGE\_FREQ / SAMPLING\_RATE;

// Configure DAC pin

PINSEL1 |= 0x00080000; // Select DAC function for P0.25

while(1){

for(sample = 0; sample < SAMPLING\_RATE; sample++){

// Generate carrier and message signals

carrier = sin(carrier\_phase);

message = sin(message\_phase);

// Generate DSBSC signal by multiplying carrier and message

dsbsc = carrier \* message;

// Scale and shift the signal to fit DAC range (0 to 1023)

DACR = ((uint32\_t)((dsbsc + 1.0) \* 511.5)) << 6;

// Increment phases

carrier\_phase += carrier\_step;

if(carrier\_phase >= 2 \* PI) carrier\_phase -= 2 \* PI;

message\_phase += message\_step;

if(message\_phase >= 2 \* PI) message\_phase -= 2 \* PI;

// Delay to achieve the sampling rate

delay\_us(1000); // Delay to achieve a 1 kHz sampling rate

}

}

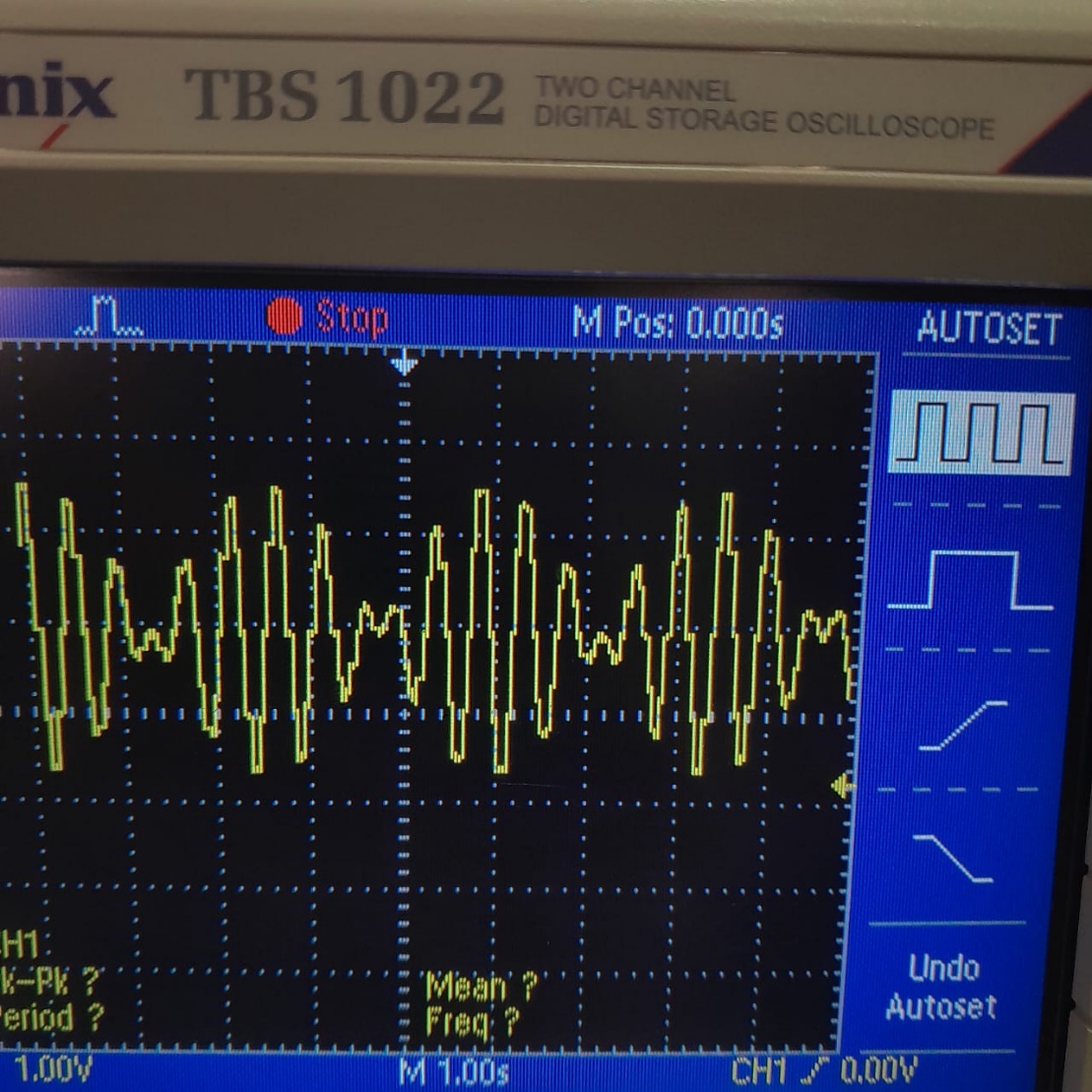
}

**Explanation:**

* Header Files: Includes necessary headers for LPC2148 definitions, standard integer types, and math functions.
* Constants: Defines PI, SAMPLING\_RATE, CARRIER\_FREQ, and MESSAGE\_FREQ for signal generation.
* Delay Function: A function to create a rough microsecond delay.
* Main Function: The main loop for generating and outputting the DSB-SC signal.
* Phase Initialization: Initializes phases for the carrier and message signals.
* Step Calculation: Calculates phase increments based on the sampling rate and signal frequencies.
* DAC Configuration: Configures the DAC pin (P0.25) for output.
* Signal Generation Loop: Continuously generates the carrier and message signals, calculates the DSB-SC signal, and outputs it through the DAC.

**5. Procedure to Create Hex File on IAR Embedded Workbench**

1. Install IAR Embedded Workbench: Ensure IAR Embedded Workbench is installed and configured for ARM development.
2. Create a New Project:
   * Open IAR Embedded Workbench.
   * Go to Project > Create New Project....
   * Select ARM and Empty Project, and click OK.
3. Add Source File:
   * Create a new C file or add an existing C file with the provided code.
   * Save the file in the project directory.
4. Configure Project Settings:
   * Go to Project > Options.
   * Under General Options, set the device to LPC2148.
   * Configure the compiler, linker, and other settings as needed.
5. Build Project:
   * Go to Project > Build All.
   * Ensure there are no errors in the code.
6. Generate Hex File:
   * Go to Project > Options.
   * Under Output Converter, check the option to generate an Intel Hex file.
   * Click OK and rebuild the project. The hex file will be generated in the project directory.
7. **Testing and Results**

* Connect the Oscilloscope: Connect the DAC output (P0.25) to an oscilloscope.
* Observe the Signal: Verify the DSB-SC waveform on the oscilloscope. The waveform should exhibit characteristics of the modulated signal.
* Analyze: Perform an FFT analysis using the oscilloscope to confirm the absence of the carrier and the presence of sidebands.

1. **Conclusion**

The LPC2148 microcontroller can successfully generate a DSB-SC signal using its DAC. The provided code demonstrates the steps needed to configure the microcontroller and output the modulated signal. By following the outlined procedures, one can create and test a DSB-SC signal, ensuring the system's efficiency and functionality for communication applications.

1. **References**
   1. LPC214x User Manual: Detailed documentation on the LPC2148 microcontroller.
   2. ARM7TDMI Technical Reference Manual: Technical details about the ARM7 core used in the LPC2148.
   3. Digital Signal Processing Textbooks: Comprehensive guides on signal processing techniques and theories.