Proposed Ultrasonic Radar Hardware Implementation with Arduino Microcontroller Based on New Techniques of Compressive sensing *using (Irregular Low Density Parity Check Code)ILDPC to Generate Measurement Matrix Φ=H*

Asst. Lact. Ali Jalil Taher
Dept. of Computer Eng,
College of Information Technology.
Computer Network and Digital Communication
University of Imam Jafar Al-Sadiq, Mayssan, Iraq
alijalil2010@yahoo.com
alijalil@cmitfod.com

Asst Prof.Dr . Ahmed Abdulsahib Hashim
Electronics and communication engineering.
College of business Information.
University of Information Tech. and Communications.
Baghdad, Iraq
dreng.ahmad@yahoo.com

Asst Prof. Dr.Thamir Rashed. Saeed
FPGA System Design and Pattern Recognition Group
Dept. of Electrical Eng.
University of Technology
Baghdad, Iraq
thamer_rashed@yahoo.com
thamir_rashed@uotechnology.edu.iq

Abstract: The design and Implementation using Arduino UNO microcontroller as simple practical Technique. The designed ultrasonic Radar performance take its comparison with respect standard using MATLAB. The proposed implementation show enhancement in system performance with on new Techniques of Compressive sensing. *using (Irregular Low Density Parity Check* ***Code)ILDPC to Generate Measurement Matrix Φ=H*** .

1-Introduction:

In this paper we introduce the hardware implementation of ultrasonic Radar and process the transmitted and echo signal by using new approach of CS by using deterministic method of get Phi = $\boldsymbol{\Phi}_{M\times N} = \boldsymbol{H}_{M\times N}$ =***ILDPC*** code to ge

2. Compressive Sensing Implementation. (CS)

CS theory enables the implementation of the recovery of the high dimensional signal from lesser observations as a comparison to the actual number of measurements required in conventional techniques. The objective of CS recovery algorithms is to provide an estimate of the original signal from the captured measurements. It is based on the property of the signals to be able to offer their representation in the sparse domain with fewer numbers of nonzero coefficients. This property is called Sparsity and the given signals as sparse signals. The reconstruction algorithm used with CS decides the number of samples needed for exact reconstruction. The model of reconstruction using CS depends on two properties: *Sparsity and Incoherence.* where can see the difference between the traditional method and compressive sensing in the figure 1 and 2 Respectively.

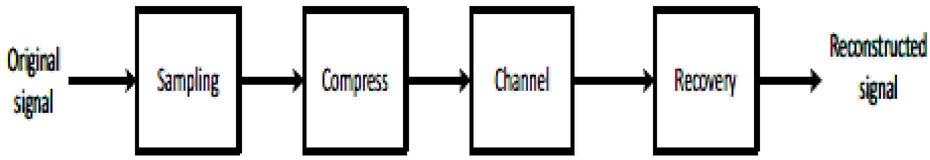


Figure 1. Conventional Signal Sampling Process.

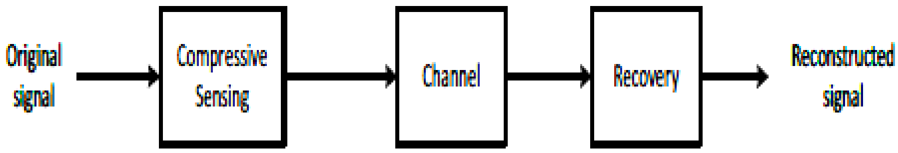


Figure 2. Signal Sampling Process Based on CS.

3-Sparsity .

Many signals are capable to be stored in compressed form in terms of their projection in a suitable basis. The projected coefficients of these signals can be zero or a far lower value, if a suitable basis is used. For a signal having non-zero coefficients, it is called -sparse. As, these sparse signals may offer the larger number of smaller coefficients that can be ignored easily; hence a compressed signal can be obtained from the sparse form. For compressive Sensing, the suitable domains available are DCT, DWT, and Fourier Transform [7]. Discrete Wavelet Transform is usually preferred over Discrete Cosine Transform because it enables the removal of blocking artifacts [8]. Basically, Sparsity refers to the possibility of having a much smaller information rate for a continuous time signal as a comparison to the one depicted by its bandwidth. So, CS can use the advantage of using these natural signals with their compressed form in a particular domain. Suppose a signal can be represented in a suitable Orthonormal basis like wavelet, DCT. As in a signal, many coefficients are small and most of the important information lies in few larger coefficients. Hence, it can be expanded in an Orthonormal basis for sparse representation. Let $\boldsymbol{x}$ the given signal and $\boldsymbol{\psi}=\{\psi_1\ \psi_2\ \psi_3\ \ldots..\psi_n\}$ represents the suitable basis, therefore, an image $\boldsymbol{x}$ in domain is given as $\boldsymbol{\psi}$ represents the suitable basis, therefore, an image in domain is given as:

$$s(t) = \sum_{i=1}^{n} x_i \, \psi_{i(t)} \quad \ldots\ldots\ldots\ldots\ldots(1)$$

Where S is the coefficients of the sparse form of x, $x_i = < s_i, \psi_i >$. In a sparse representation of the signal, small coefficients in that signal can be neglected without much information loss. It's like considering the signal by keeping only the significant coefficients and discarding the smaller coefficients. Thus, the obtained vector is known as a sparse signal see Figure.3.

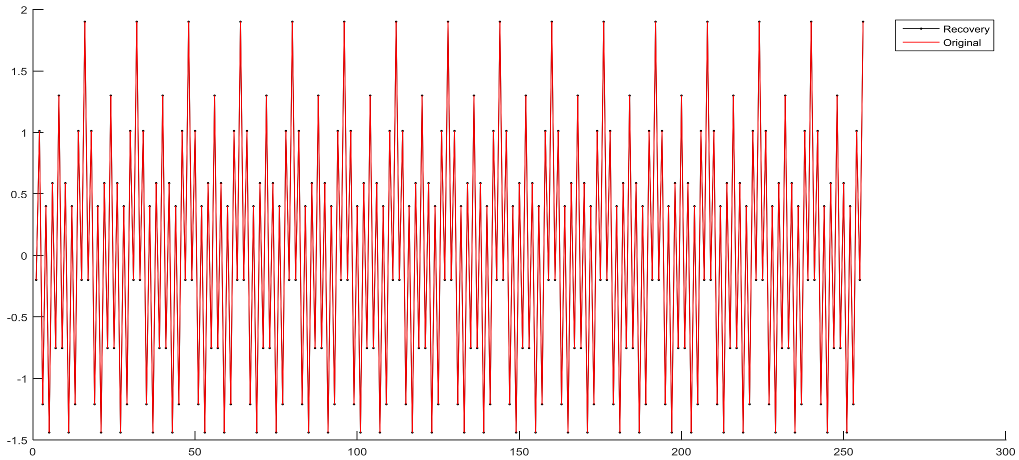


Figure.3. ***Recovery of the original signal by using Compressive sensing N=256.***

4-Incoherence.

Incoherence shows that any signal with a sparse representation in a particular domain can be spread out in a domain in which it is actually captured. It enables the relationship of duality between time and frequency. It measures the maximum correlation for any two matrices. These two matrices give a form of different representation domains. For the measurement matrix Φ with size $M \times N$ and the representation matrix $\Psi_{N\times N}$ of size $N \times N$, the representation matrix can be represented as Ψ_1 … Ψ_n as columns and measurement matrix as Φ_1…… Φ_n as rows. The coherence is given as.

$$\mu(\Phi, \Psi) = \sqrt{n} \max \left| \Phi_k , \Psi_j \right| \quad \ldots\ldots(2)$$

for $1 \le j \le n$, and $1 \le k \le m$. Moreover, from linear algebra, for incoherence following result can be depicted.

$$1 \le \mu(\Phi, \Psi) \le \sqrt{n} \quad \ldots\ldots\ldots\ldots\ldots(3)$$

In CS technology, the incoherence of two matrices is important. One is the Sensing matrix that is used to sense the significant columns of the signal of interest. The second one is the representation matrix $\Psi_{N\times N}$ in which the given signal is represented in the sparse form. The low value of incoherence for CS shows that the fewer measurements are required for reconstruction of the signal [9].

Coherence is able to measure the maximum correlation between the columns or elements of $\boldsymbol{\Psi}_{N\times N}$ and $\boldsymbol{\Phi}$. Mostly low coherence pairs are considered in Compressive Sensing. The measurement matrix $\boldsymbol{\Phi}$ basically performs the function of sampling the coefficients. The measurement matrices like Fourier, Gaussian are able to satisfy the coherence property. The random matrices like i.i.d (independent identically distribution)Gaussian Matrix or binary ±1 matrix with fixed basis $\boldsymbol{\Psi}_{N\times N}$ are mainly incoherent. These matrices are simple and possess lower convergence, which are required for recovery with Compressive Sensing.

5- <u>Compressive Sensing Model .</u>

Compressive Sensing model basically performs compression and sampling simultaneously. Considering an $\boldsymbol{N}$ dimensional signal , the sparse form of the signal can be constructed by representing it in any suitable basis like DCT, Fourier Transform, and wavelet Transform. The sparse form or the signal of interest can be given as:

$\boldsymbol{x} = \boldsymbol{\Psi}_{N\times N}\boldsymbol{s}$ ………..(4)

Where $\boldsymbol{x}$ is the sparse form of $\boldsymbol{x}$ to $\boldsymbol{s}$ and $\boldsymbol{\Psi}_{N\times N}$ is the suitable basis that shows the projection coefficients of $\boldsymbol{x}$ on the given basis. The next step is to compute the measurement vector $\boldsymbol{y}$ with a suitable matrix either Gaussian [12] or Bernoulli [13]. The measured vector can be given as:

$\boldsymbol{y} = \boldsymbol{\Phi}_{M\times N}\,\boldsymbol{x}$ ………..(5)

where $\boldsymbol{\Phi}_{M\times N}$ is the measurement matrix of dimension $\boldsymbol{M\times N}$. The overall Eq. can be represented as:

$\boldsymbol{y}= \boldsymbol{\Theta}_{M\times N}\,\boldsymbol{s}$ ………..(6)

Where $\boldsymbol{\Theta}_{M\times N}$ is the Sensing matrix and is depicted as $\boldsymbol{\Theta}_{M\times N} = \boldsymbol{\Phi}_{M\times N}\,\boldsymbol{\Psi}_{N\times N}$, it is also known as reconstruction matrix. In practical applications, the measurement or the random noises can also be considered. Equations (5) and (6) are reformulated as: Figure 3.

$\boldsymbol{y} = \boldsymbol{\Phi}_{M\times N}\,\boldsymbol{x} + \boldsymbol{e}$………..(9)

$\boldsymbol{y}= \boldsymbol{\Theta}_{M\times N}\,\boldsymbol{s} + \boldsymbol{e}$ ………..(10)

Where $\boldsymbol{e}$ represents Random noise vector. Hence, the primary objective of CS is to recover the signal from these captured measurements, under sparsifying conditions. Then, the recovery algorithms are applied on the given measurement vector. The recovery algorithms available are $\boldsymbol{L}_1$ minimization.

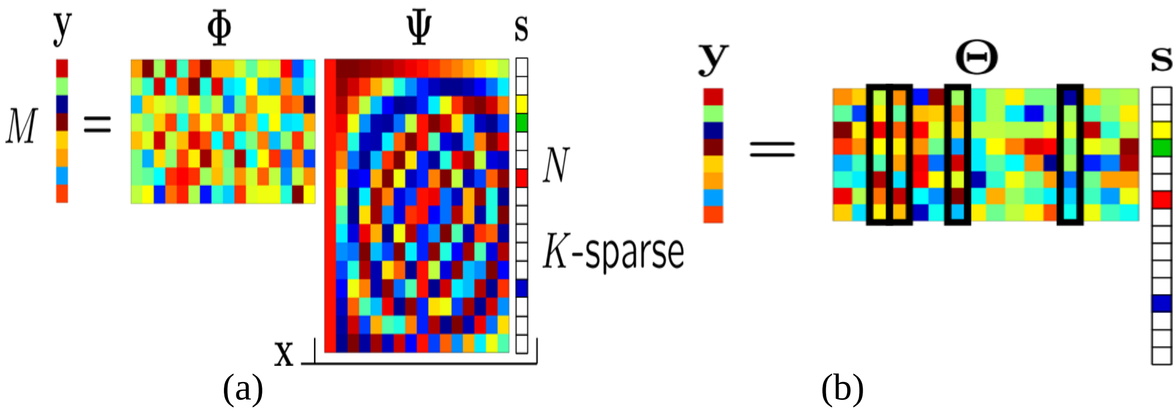


Figure 4: ***(a) Compressive sensing measurement process with (random Gaussian) measurement matrix $\Phi_{M\times N}$ and Discrete Cosine Transform (DCT) matrix $\Psi_{N\times N}$. The coefficient vector s is sparse with K = 4. (b) Measurement process in terms of the matrix product $\Theta_{M\times N} = \Phi_{M\times N}\ \Psi_{N\times N}$ with the four columns corresponding to nonzero s_i highlighted. The measurement vector y is a linear combination of these four columns.***

6-LDPC (Low Density Parity Check code) H.

6-1 Graphical Representation.

Tanner considered ***LDPC*** codes (and a Generalization) and showed how they may be represented effectively by a so-called bipartite graph, now call graph [13]. The Tanner graph of an LDPC code is analogous to the trellis of a convolutional code in that it provides a complete representation of the code and it aids in the description of the decoding algorithm .A bipartite graph is a graph (nodes connected by edges) whose nodes may be separated into two types and edges may only connect two nodes of different types. The two types of nodes in Tanner graph are the variable nodes and the check nodes (which we shall call v-nodes and c-nodes, respectively).(the nomenclature varies in the literature : variable nodes are also called bit or symbol nodes and check nodes are also called function nodes) The Tanner graph of a code is drawn according to the following rule: check node j is connected to variable node i whenever element h_{ij} in H is 1. One deduce from this that there are m=n-k check nodes, one for each check equation , and n variable nodes ,one for each code bit c_i .Further , the m rows of H specify the m c-node connections, and the n columns of H specify the n v-node connections.

Example. Consider a (10,5) linear block code with w_c =2 and w_r = w_c(n/m) =4 with the following H. Matrix .

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

The Tanner graph corresponding to $\boldsymbol{H}$ is depicted in fig 1. Observe that ***v***-nodes $\boldsymbol{c}_0,\boldsymbol{c}_1,\boldsymbol{c}_2$ and $\boldsymbol{c}_3$ are connected to ***c***-node $\boldsymbol{f_0}$ accordance with the fact that , in the zeroth row of $\boldsymbol{H}$,$\boldsymbol{h_{00}}$= $\boldsymbol{h_{01}}$= $\boldsymbol{h_{02}}$= $\boldsymbol{h_{03}}$= 1 (all others are zero).Observe that analogous situations holds for ***c***-nodes $\boldsymbol{f_1}$, $\boldsymbol{f_2}$, $\boldsymbol{f_3}$, and $\boldsymbol{f_4}$ which corresponds to rows 1,2,3 and 4 of $\boldsymbol{H}$, respectively. Note m as follows from the fact that $\boldsymbol{cH^T}$ =0. The bit values connected to the same check node must sum to zero .we may also proceed along columns to construct the Tanner graph. For example , note that ***v***-node $\boldsymbol{c_0}$ is connected to ***c***-nodes $\boldsymbol{f_0}$ and $\boldsymbol{f_1}$ in accordance with the fact that , in the zeroth column of $\boldsymbol{H}$, $\boldsymbol{h_{00}}$= $\boldsymbol{h_{10}}$=**1.**

Note that the Tanner graph in this example is regular: each v-node has two edge connections. And each ***c***-node has four edge connections (that is, the degree of each ***v***-node is 2 and degree of each ***c***-node is 4). This is in accordance with the fact that $\boldsymbol{w_c}$ =**2** and $\boldsymbol{w_r}$ =**4**. Is also clear from this example that $\boldsymbol{mw_r=nw_c}$. For Irregular ***LDPC*** codes , the parameters $\boldsymbol{w_c}$ and $\boldsymbol{w_r}$ are functions of the column and row numbers and so such notation is not generally adopted in this case . instead m it is usual in the literature [7] to specify the ***v***-node and ***c***-node

degree distribution polynomials., denoted by $\boldsymbol{\lambda}(\boldsymbol{x})$ and $\boldsymbol{p}$(x), respectively . in the polynomial.

$$\lambda(\boldsymbol{x}) = \sum_{d=1}^{dv} \lambda_d \, x^{d-1} \quad ,$$

$\boldsymbol{\lambda_d}$ denotes the fraction of all edges connected to degree d. v-nodes and $\boldsymbol{d_v}$ denotes the maximum v-node degree . Similarly , in the polynomial.

$$\boldsymbol{p}(\boldsymbol{x}) = \sum_{d=1}^{dc} p_d \, x^{d-1} \quad ,$$

$\boldsymbol{p_d}$ denotes the fraction of all edges connected to degree d c-nodes and $\boldsymbol{d_c}$ denotes the maximum c-node degree. Note for the regular code above , for which $\boldsymbol{w_c}$ =$\boldsymbol{d_v}$=2 and $\boldsymbol{w_r}$ =$\boldsymbol{d_c}$=4, we have $\boldsymbol{\lambda}(\boldsymbol{x})$=$\boldsymbol{x}$ and $\boldsymbol{p}(\boldsymbol{x})$=x ^{3}.

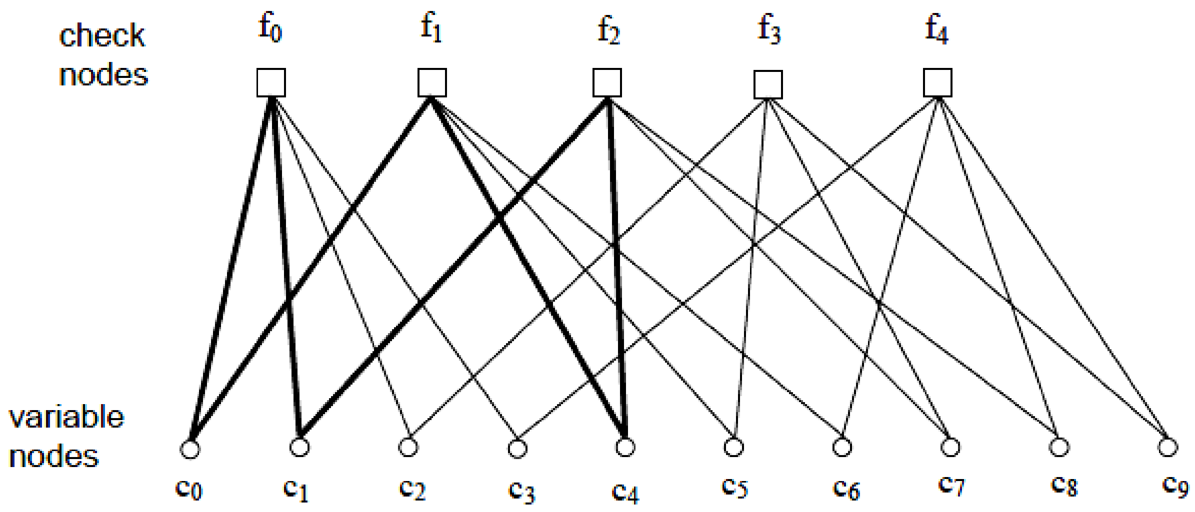


Figure (5) Tanner graph for example code.

A cycle (or loop) of length ***v*** in a Tanner graph is a path comprising ***v*** edges which closes back on itself. The Tanner graph in the above example posses a length -6 cycle as exemplified by the six bold edges in the figure. The girth γ of a Tanner graph is the minimum cycle length of the graph. the shortest possible cycle in bipartite graph is clearly a length-4 cycle, and a such cycles manifest themselves in the ***H*** matrix as four **1**s that lie on the corners of a submatrix of ***H*** .we are interested in cycles, particularly short cycles , because they degrade the performance of the iterative decoding algorithm used for LDPC codes .

6-2.LDPC Code Design Approaches.

Clearly , the most obvious path to the construction of LDPC code is via the construction of a low-density parity-check matrix with prescribed properties. A large number of design techniques exist in the literature , and we introduce some of the more prominent ones in this section , albeit at a superficial level. The design approaches target different design criteria , including efficient encoding and decoding ,near –capacity performance ,or low-error rate floors. (Like turbo codes, LDPC codes often suffer from low-error rate floors, Owing both to poor distance spectra and weaknesses in the iterative decoding algorithm) .

6-3.Gallager Codes.

The original LDPC codes due to Gallager [13] are regular LDPC codes with n H matrix of the form.

$$H = \begin{bmatrix} H_1 \\ H_2 \\ \vdots \\ H_{w_c} \end{bmatrix}$$

Where the sub matrices H_d have the following structure . From any integers μ and w_r than greater than 1 .each submatrix H_d is $\mu * \mu w_r$ with row weight w_r and column weight 1.the submatrix.

H_1 has the following specific form : for i=1,2,3,….., μ the ***i***-th row contains all of its w_r 1***'s*** in columns **(*i*-1)** w_r**+1** to iw_r .the other submatrices are simply column permutations of H_1 . its evident that ***H*** is regular has dimension $\mu w_c * \mu w_r$, and has row and column weights w_r and w_c , respectively . the absence of length-4 cycles in H is not guaranteed ,but they can be avoided by computer design of ***H*** .Gallager showed that the ensemble of such codes has excellent distance properties provided $w_c \geq 3$ and $w_r > w_c$ .Further Such codes have low-Complexity encoders since parity bits can be solved for as a function of the user bits via the parity-check matrix [13].Gallager codes were generalized by Tanner in 1981[16]. And were studied for application to code-division multiple- access (CDMA). Channel in[20].Gallager codes were extended by MacKay other.[14],[15].

6-4.MacKay. Codes.

MacKay had independently discovered the benefits of the designing binary codes with sparse ***H*** matrices and was the first to show the ability of these codes to perform near capacity limits[13],[14]. MacKay has archived on a web page[10] a large number of LDPC codes has designed for application to data communication and storage, most of which are regular .He provided in [4].algorithm to semi-randomly generate sparse ***H*** matrices. A few of these are listed below in order of increasing algorithm complexity (but not necessarily improved performance).

1- ***H*** is created by randomly generating weight-w_c columns and (as near as possible) uniform row weight.
2- ***H*** is created by randomly generating weight -w_c column ,while ensuring weight-w_r rows and no two columns having overlap greater than one.
3- ***H*** is generated as in 2,plus short cycles are avoided.
4- ***H*** is generated as 3,plus $H = [H_1 \ \ H_2]$ is constrained so that H_2 is invertible(or at least ***H*** is full rank).one drawback of Mackay Codes is that they lack sufficient structure to enable low-complexity encoding .Encoding is performed by putting ***H*** in the form $[P^T \ I]$ via Gauss-jordan elimination,

From which the generator matrix can be put in the systematic form $G=[I \ \ P]$. the problem with encoding via ***G*** is the submatrix ***P*** is generally not sparse so that ,for codes of length $\boldsymbol{n}$=100 or more encoding complexity is high .An efficient encoding technique employing only the $\boldsymbol{H}$ matrix was proposed in [16].

6-5. Irregular LDPC codes .

Richardson et al.[17] and Luby et al[18] defined ensembles of irregular LDPC codes parameterized by the degree distribution polynomials $\lambda(x)$ and $p(x)$ and showed how to optimizes these polynomials for a variety of channels. Optimality is in the sense that ,assuming message-passing decoding (described below),a typical code in the ensemble was capable of reliable communication in worse channel conditions than codes outside the ensemble are .the worst-case channel condition is called the decoding threshold and optimization of $\lambda(x)$ and $p(x)$ is found by a combination of a so-called ***density evolution*** algorithm and an optimization algorithm. Density

Evolution refers to the evolution of the probability density functions(pdfs) of the various quantities passed around the decoders Tanner graph. The decoding threshold for a given $\lambda(x)$ - $p(x)$ pair is determined by evaluation the pdfs of computed log-likelihood ratios of the code bits .the spate optimization algorithm optimizes over the $\lambda(x)$ - $p(x)$ pairs.

Using this approach an irregular ***LDPC*** code has been designed whose simulated performance was within 0.045dB. of the capacity limit for a binary-input AWGN channel[15].This code had length $\boldsymbol{n}=\mathbf{10^7}$ and rate $\boldsymbol{R}$=1/2 . it is generally true that designs via density evolution are best applied to codes whose rate is not too high ($\boldsymbol{R} \leq 3/4$) and whose length is not too short ($\boldsymbol{n} \geq \mathbf{5000}$) . The reason is that the density evolution design algorithm assumes $\boldsymbol{n} \rightarrow \infty$(hence ,$\boldsymbol{m} \rightarrow \infty$) and so $\lambda(x)$ - $p(x)$ pairs which are optimal for very long codes, will not be optimal for medium-length and short codes. As discussed in [15],[16],[17],[18], such $\lambda(x)$ - $p(x)$ pairs applied to medium-length and short codes gives rise to a high error floor. Finally, we remark that ,as for the MacKay codes, these irregular codes do not intrinsically lend themselves to efficient encoding. However ,as mentioned above ,Richardson and Urbanke [16]. Have Proposed algorithms for achieving linear-time encoding for these codes.

Table 1. **Arduino UNO technical specification** [20].

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

To overcome memory limitation of Arduino UNO, SD card used with suitable connection with Arduino UNO. The connection between them listed in Table 3.

Table 2. SD **card connected with Arduino UNO** [22].

GND: ground (0V)
VCC: power supply (5V)
MISO (Master Input Slave Output) connected to Arduino UNO pin 12
MOSI (Master Output Slave Input) connected to Arduino UNO pin 11
SCK (Master Serial Clock) connected to Arduino UNO pin 13
CS (Slave Select) hint : you can connect this pin to any Arduino digital output

Table 3. **Ultrasonic Ranging Module HC - SR04.[21]**

Working Voltage	DC 5 V
Working Current	15mA
Working Frequency	40Hz
Max Range	4m
Min Range	2cm
MeasuringAngle	15 degree
Trigger Input Signal	10uS TTL pulse
Echo Output Signal	Input TTL lever signal and the range in proportion
Dimension	45*20*15mm

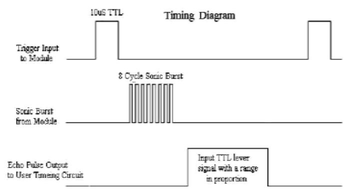
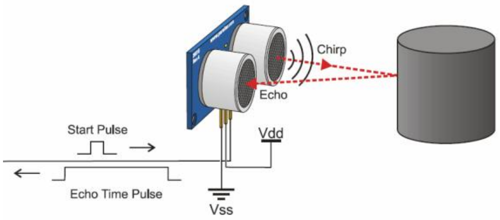
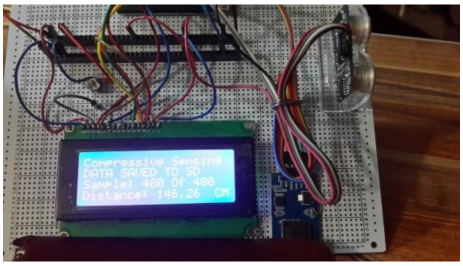


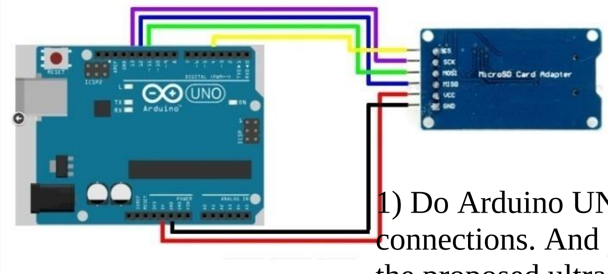
Table 4. **Wire Connection of the Bored of ultrasonic Radar using Arduino Uno..**

color Wire	*Arduino Uno*	*SD CARD*	*Ultrasonic Ranging Module HC - SR04.*	*Monitor screen*
	Rx-0	**X**	**X**	**X**
	Tx-1	**X**	**X**	**X**
Red	**Digital-2**	**X**	**X**	**RS**
Blue	**Digital-3**	**X**	**X**	**E**
Green	**Digital-4**	**X**	**X**	**D4**
Yellow	**Digital-5**	**X**	**X**	**D5**
Orange	**Digital-6**	**X**	**X**	**D6**
Red	**Digital-7**	**X**	**X**	**D7**
Gray	**Digital-8**	**X**	**Trig**	**X**
white	**Digital-9**	**X**	**Eco**	**X**
Brown	**Digital-10**	**CS**	**X**	**X**
Orange	**Digital-11**	**MOSI**	**X**	**X**
Yellow	**Digital-12**	**MISO**	**X**	**X**
Red	**Digital-13**	**SCK**	**X**	**X**
Red	**5V**	**X**	**X**	**VDD**
Gray	**5V- Potential-resistance.**	**X**	**X**	**VO**
Red	**5V**	**X**	**X**	**A**
Red	**5V**	**X**	**VCC**	**X**
Green	**5V**	**VCC**	**X**	**X**
Blue	**GND**	**GND**	**X**	**X**
Blue	**GND**	**X**	**X**	**K**
BLACK	**GND**		**GND**	
Blue	**GND –Potential Resistance**			

The Arduino UNO micro controller after program algorithm, it uses serial port to transmit data to user interface and save the calculated permuted sequence as text file in SD card. Where the transmitted signal and received its real ultrasonic signal using frequency 40 kHz. The stored information in text file will be exported to MATLAB, by using SD card. a full simulation function in M-file that simulate communication system will import the stored information and use it to calculate system performance and plot the results as comparison between system using standard and convolution CS- and proposed CS- Using Irregular ***ILDPC*** codes to generate measurement $\boldsymbol{\Phi}$ *matrix* . the process illustrated in Figure 6.

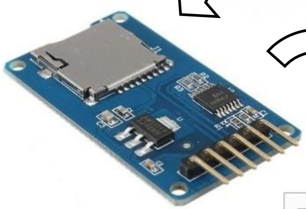




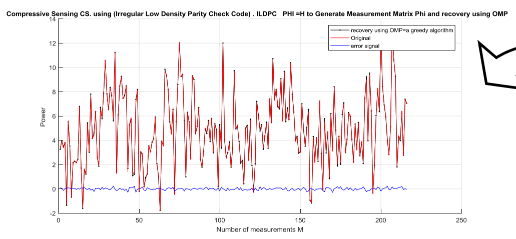


1) Do Arduino UNO connections. And program the proposed ultrasonic radar sensor HC-SR4 algorithm

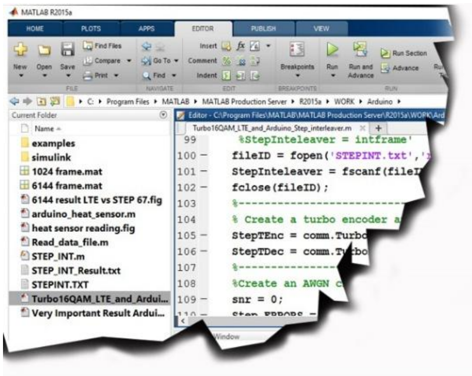
2) Transfer ultrasonic radar result Through serial port



3) Create and write ultrasonic results sequence to text file



5) MATLAB import the text file and use its data through communication system m files, test



4) Save text file to SD card memory and export it to MATLAB

Figure 6. Illustrate Proposed Ultrasonic Radar Hardware Implementation with Arduino Micro Controller Based on compressive sensing CS. Using ***ILDPC*** to generate measurement matrix $\boldsymbol{\Phi}_{M\times N}$.

7-Simulation Results

In this study we can compare the results by using conventional methods of compressive sensing recovery Algorithms Figure 7.($\boldsymbol{\Phi}_{M\times N}$.= Random Gaussian or Bernoulli Matrix) and the Figure 9. Compare the results by using methods of Using The Proposed New techniques Of compressive sensing recovery Algorithms by Using ***ILDPC*** to generate Measurement matrix $\boldsymbol{\Phi}_{M\times N}$. $=\boldsymbol{H}_{M\times N}$. Which is deterministic method (not random) we can see the recovery all the time for all algorithms is 99%. Recovery.

Figures 7,8 and 9 show simulation results.

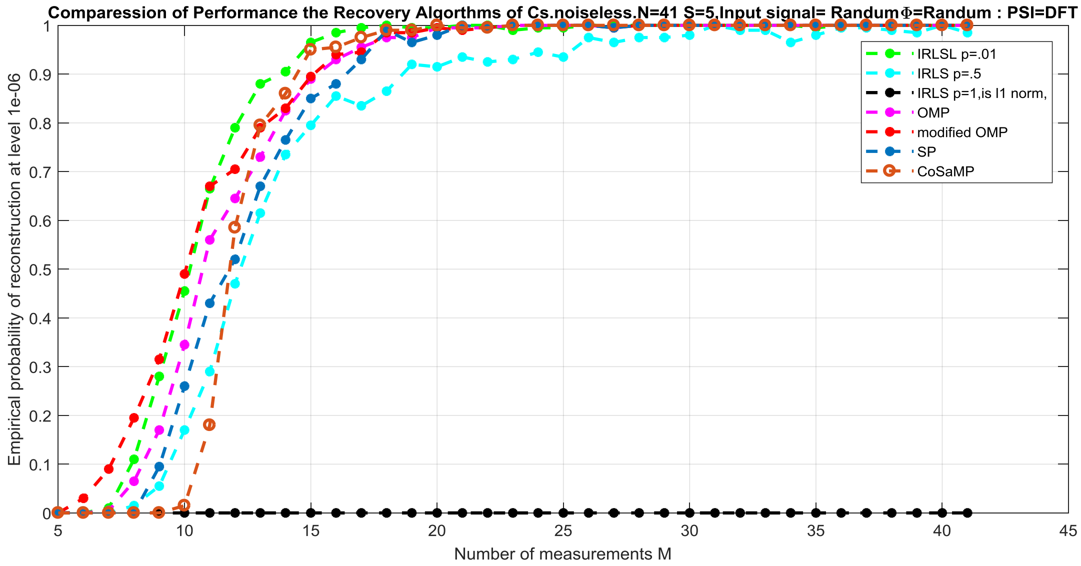


Figure 7. Compression the Performance of the Recovery Algorithms Of the Conventional Compressive sensing CS By using $\boldsymbol{\Phi}_{M\times N}$.=Random Gaussian or Bernoulli.

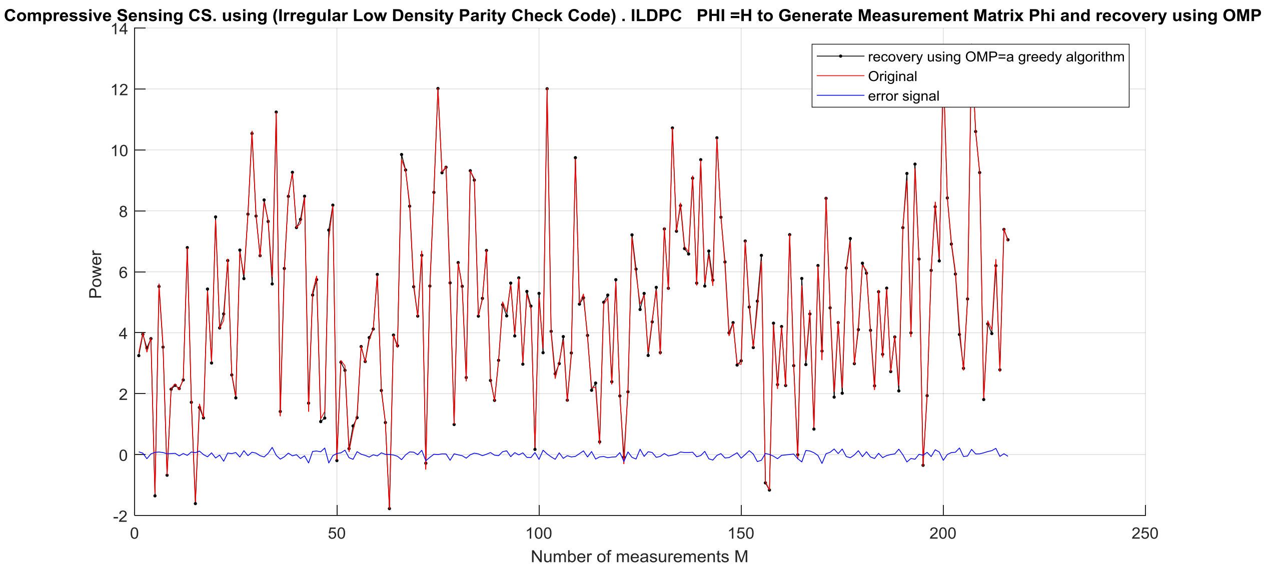


Figure.8**. *Recovery of the original signal by using New techniques of Compressive sensing ILDPC-***

CS with Ultrasonic Real Signal. N=256.

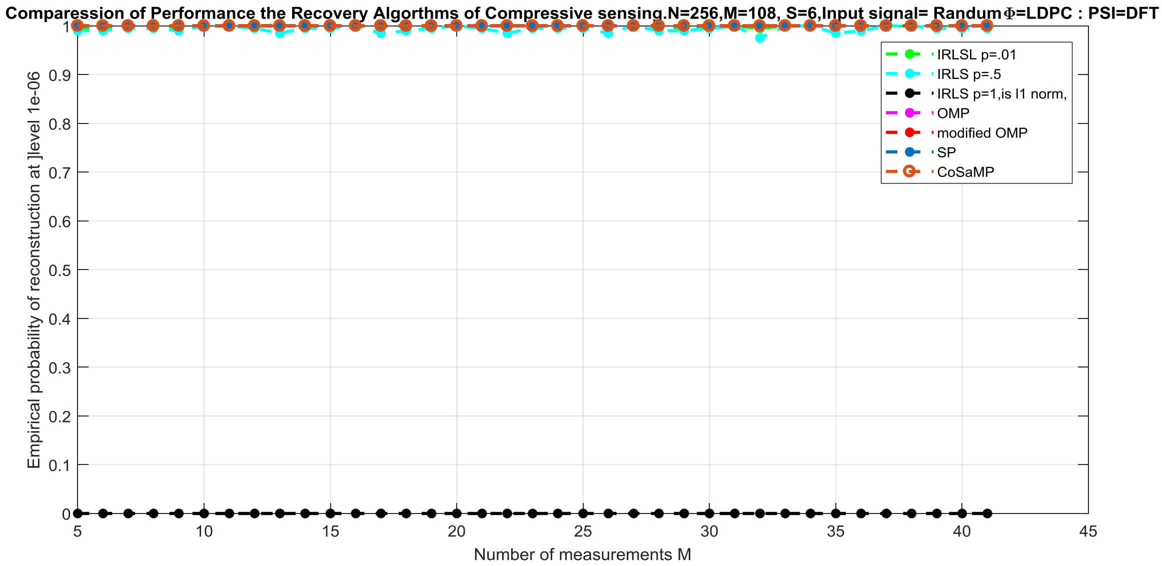


Figure 9. Compression the Performance of the Recovery Algorithms Of the Compressive Sensing CS. By Using $\boldsymbol{\Phi_{M\times N}}$.=*ILDPC*.

8-Conclusions

Implement proposed Ultrasonic Real signal using Arduino UNO with sensor HCSR04- Ultrasonic sensor for implementation of new techniques of compressive sensing by using ILDPC(Irregular Low Density Parity-Check Code. Which is deterministic technique (Not Random) to generate measurement matrix $\boldsymbol{\Phi_{M\times N}}$. =$\boldsymbol{H_{M\times N}}$. Instead Of convolution method by using $\boldsymbol{\Phi_{M\times N}}$. = Random Gaussian or Bernoulli Random variable .we see at figure 7 we need more Measurements ***M*** for all types of recovery Algorithms ,Subspace Pursuit(SP), Orthogonal Matching Pursuit (OMP), **(*modifiedOMP*)** ,Iterative Reweighted Least Square Algorithm (IRLS).and The Compressive Sampling Maching Parsuite Algorithm (CoSaMP).
The recovery of the received signal approximately 99% . For new the techniques as $\boldsymbol{\Phi_{M\times N}}$. = $\boldsymbol{H_{M\times N}}$. as compare with the conventional one by using $\boldsymbol{\Phi_{M\times N}}$. = Random Gaussian or Bernoulli Random variable . we see more measurements M to recovery the received signal for the same Algorithms to get the same transmitted signal . Illustrated in figure 7.

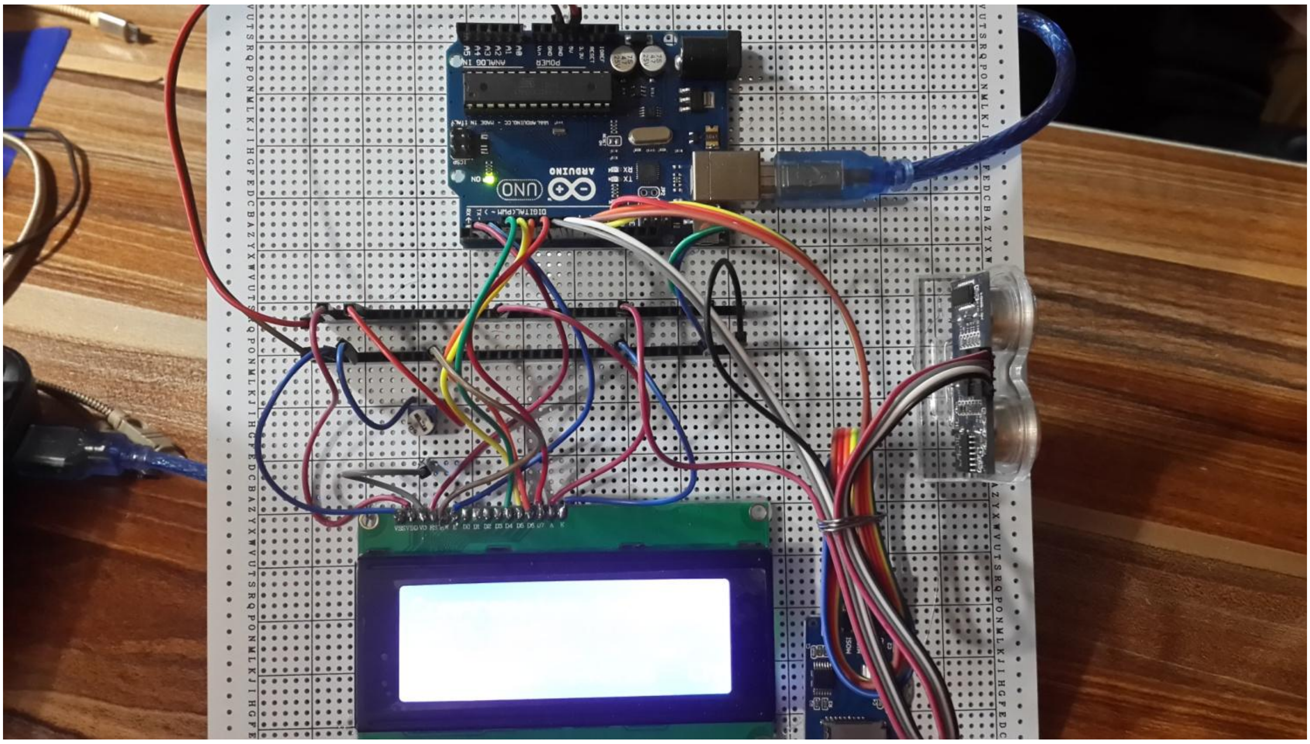
References:

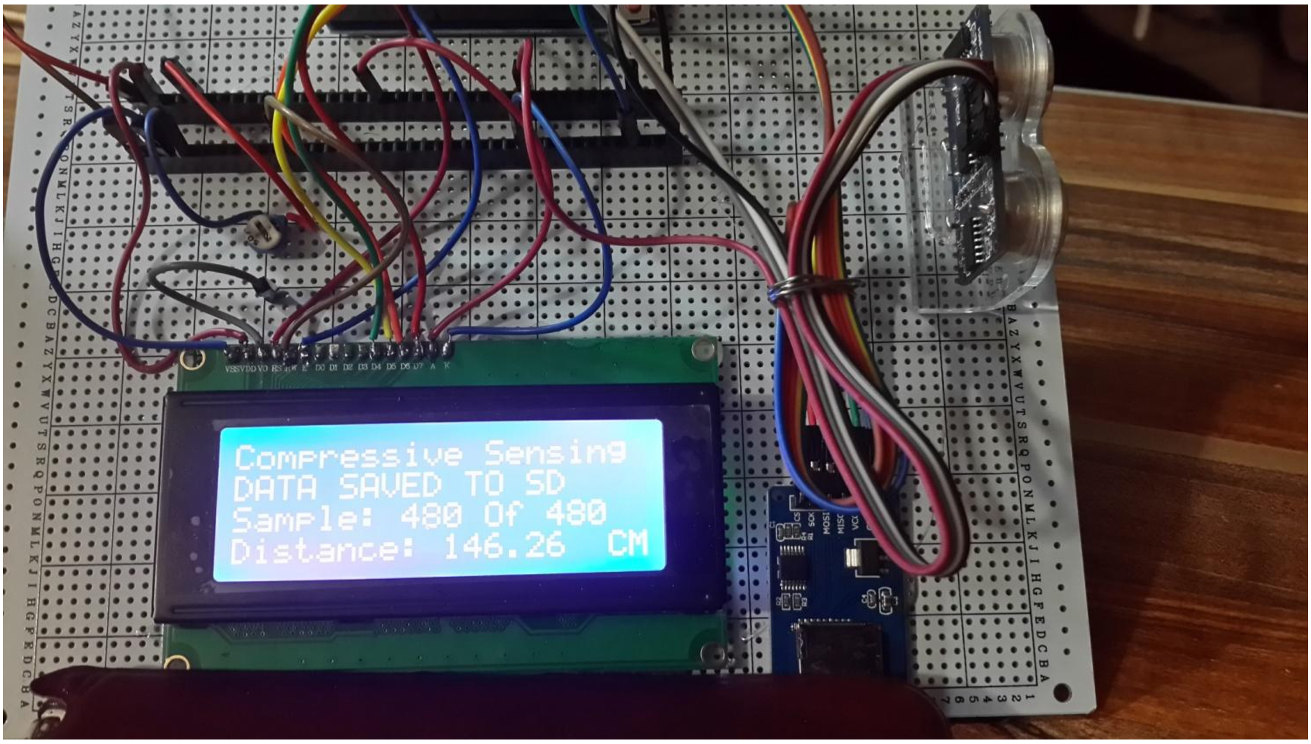
[1] D. L. Donoho, “Compressed Sensing,” *IEEE Trans. Inform. Theory*, vol. 52, no. 4, pp. 1289-1306, 2006.
[2] E. J. Candes and M. Wakin, “An Introduction to Compressive Sampling,” *IEEE Signal Process. Magazine*, vol. 25, no. 2, pp.21- 30, March 2008.

[3] E. J. Candes, "Compressive Sampling," *In Proceedings of the International Congress of Mathematicians*, pp. 1433-1452, Madrid, Spain, 2006.
[4] E. J. Candes, J. Romberg and T. Tao, "Robust Uncertainty Principles: Exact Signal Reconstruction from Highly Incomplete Frequency Information," *IEEE Trans. Inform. Theory*, vol. 52, no. 2, pp. 489-509, February 2006.
[5] E. J. Candes, J. Romberg and T. Tao, "Stable Signal Recovery from Incomplete and Inaccurate Measurements," *Comm. Pure Appl. Math.*, vol. 59, no.8, pp. 1207-1223, 2006.
[6] E. J. Candes and J. Romberg, "Sparsity and Incoherence in Compressive Sampling," *IEEE signal processing magazine*, vol. 23, no. 3, pp. 969-985, 10th April 2007.
[7] E. J. Candes and T. Tao, "Reflection on Compressive Sensing," *IEEE signal processing magazine*, vol. 58, no. 4, pp. 14-18, December 2008.
[8] R. Baraniuk, "Compressive Sensing," *IEEE Signal Processing magazine*, vol. 24, no. 4, pp. 118-121, July 2007.
[9] E. J. Candes and T. Tao, "Near Optimal Signal Recovery from Random Projections: Universal Encoding Strategies?," *IEEE Trans. Inform. Theory*, vol. 52, no. 12, pp. 5406-5425, 2006.
[10] R. Chartrand, "Exact Reconstruction of Sparse Signals via Nonconvex Minimization," *Signal Process. Lett.*, vol. 14, no. 10, pp. 707-710, 2007.
[11] C. L Lawson, *Contributions to the Theory of Linear Least Maximum Approximation*, PhD thesis, University of California, Los Angeles, 1961.
[12] I. Daubechies, R. DeVore, M. Fornasier, and S. Gunturk, "Iteratively Reweighted Least Squares minimization: Proof of Faster than Linear
[13]R.Gallager ,"Low-density parity-check codes," *IRE Trans .Information Theory*,PP.21-28,Jan.1962.
[14] R.M. Tanner "A recursive approach to low complexity codes," IEEE Trans. Information Theory ,pp, 533-547,Sept.1981.
[15] D. MacKay and R .Neal ,"Good codes based on very sparse matrices," in Cryptography and Coding , 5th IMA Conf., C Boyd, Ed., Lecture Notes in Computer Science, pp,100-111,Berlin,Germany :Springer ,1995.
[16] D. MacKay ,"Good error correcting codes based on very sparse matrices ,"*IEEE Trans. Information Theory* ,pp, 399-431,March.1999.
[17]N. Alon and M. Luby, "A linear time erasure-resilient code with nearly optimal recovery," *IEEE Trans Information Theory* ,pp, 1732-1736,Nov.1996.
[18]T.Richardson ,and R. Urbanke,"Efficient encoding of low-density parity-check codes," *IEEE Trans Information Theory* ,vol47,pp, 638-656,Feb.2001.
[19] T.Richardson ,A.Shokrollahi ,and R. Urbanke,"Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans Information Theory* ,vol47,pp, 619-637,Feb.2001.

[20]M. Luby, Mitzenmacher M.Shokrollahi ,and D.Spielman,"Improved low-density parity-check codes using irregular graphs," *IEEE Trans Information Theory* , ,pp, 585-598,Feb.2001.
[21] https://store.arduino.cc/usa/arduino-uno-rev3
[22] https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf
[23] https://www.arduino.cc/en/Reference/SDCardNotes.

Appendix.





4.Arduino code.

```
/*design HCSR04- ULTRASONIC SENSOR WITH ARDUINO UNO TO
DETECT OBJECTS.
BY Ph.D Student Ali Jalil Taher.
Bahgdad/ Iraq/University of Technology/ Electricaldwaz\ Dept./2017-8.
 HC-SR04 Ping distance sensor]
 VCC to arduino 5v GND to arduino GND
 Echo to Arduino pin 13 Trig to Arduino pin 12
 Red POS to Arduino pin 11
 Green POS to Arduino pin 10
 560 ohm resistor to both LED NEG and GRD power rail

 */

#define trigPin 13
#define echoPin 12
#define led 11
#define led2 10

void setup() {
  Serial.begin (9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
```

```
  pinMode(led, OUTPUT);
  pinMode(led2, OUTPUT);
}

void loop() {
  long duration, distance;
  digitalWrite(trigPin, LOW);  // Added this line
  delayMicroseconds(2); // Added this line
  digitalWrite(trigPin, HIGH);
//  delayMicroseconds(1000); - Removed this line
  delayMicroseconds(10); // Added this line
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = (duration/2) / 29.1;
  if (distance < 4) {  // This is where the LED On/Off happens
    digitalWrite(led,HIGH); // When the Red condition is met, the Green LED
should turn off
  digitalWrite(led2,LOW);
}
  else {
    digitalWrite(led,LOW);
    digitalWrite(led2,HIGH);
  }
  if (distance >= 200 || distance <= 0){
    Serial.println("Out of range");
  }
  else {
    Serial.print(distance);
    Serial.println(" cm");
  }
  delay(500);
}
```