Ventilator/Respirator Hardware and Software Design Specification

Rev. 0, 11/2011

freescale™

Contents

Section Number **Title** **Page**

Chapter 1
Introduction

Chapter 2
Features

Chapter 3
System Background

Chapter 4
Hardware Design

Chapter 5
Software Design

Chapter 6
Parameters Configuration

Chapter 7
Troubleshooting

Chapter 8
Conclusion

Chapter 1
Introduction

1.1 Introduction

The ventilator (also known as a respirator) is a pneumatic and electronics system designed to monitor, assist, or control pulmonary ventilation, and respiration intermittently or continuously. It can also be used to control human body oxygen levels, for example during surgery where blood loss can result in hypoxia, or lack of sufficient oxygen in the patient´s body; it is best to have less human interaction. Mechanical ventilation is designed to maintain an adequate exchange of gases, even through diminished breathing rates and reduced myocardial use. However, it can also be used to provide adequate lung expansion, the correct combination of anesthetic sedation for muscle relaxing, and stabilize the thoracic wall.

The respirator is made of a compressed air reservoir, air and oxygen supplies, a set of valves and tubes, and a disposable or reusable patient circuit. The air reservoir is pneumatically compressed several times a minute to deliver room-air or in most cases an air/oxygen mixture. The lungs elasticity allows releasing the overpressure, this is called passive exhalation, and the exhaled air is released usually through a one-way valve within the patient circuit. The oxygen content of the inspired gas can be set from 21 percent (ambient air) to 100 percent (pure oxygen).

This reference design simulates basic human lung behavior. It is easy to test different pulmonary therapies without connecting a lung to the device. The objective of this development platform is to showcase Freescale product capability while developing a ventilator or respirator. it represents a complex application where accurate measurement, correct instrumentation, power manager, and signal integrity are a critical factor for correct operation of a machine which a human life may depend on.

Chapter 2
Features

2.1 Features

Main components:

- MCF51MM256 32-bit Coldfire Freescale MCU with analog modules (DAC, internal Op Amps) ideal for medical and instrumentation appliances.
- MPXV5050GP (0 to 50 KPa single pressure) and MPX7002DP (-2 to +2 KPa differential pressure) compensated Freescale sensors.
- MHP1-M4H (14 lts/min, at 5 V, 1 W, 250 Hz, 2/2, M3) electro valves.
- 12 V at 10 A i300 Good Year automotive Air compressor 15 lts/min.
- 500 mA transistors power stage (TIP31C) for valves and buzzer, and a 15 A relay (JSM1A-12V-5) for the air compressor.
- Aluminum air mixture chamber (30 PSI max) 4 outputs (M3).
- Multi-gain analog temperature sensor 0-3 V output voltage.
- 20 X 4 character 5 V LCD display (C-51847NFJ-SLW-ADN).
- Alarm buzzer MAG 2.0 KHZ 3 V
- Tactile buttons and LED indicators.
- USB device connector.
- 1 L anaesthetic bag to simulate human lungs
- Medical Venturi pipe to measure the flow from the pressure difference

Functions:

- Moves air in and out of the air container to assist, monitor, or control ventilation
- Control air mixture percentage by pressure.
- Human interface to monitor and control main parameters as respiration frequency, pressure, measure units and control mode.
- Lungs basic behavior simulation (air container bag).
- Three control modes (Pressure, Frequency, and Assisted)
- Control of one air compressor, and supportable PID functions for four valves

Resources:

This system uses the following resources from the MCF51MM256:

- 11 I/O ports
- Five 16-bit ADC ports
- Three TPM ports
- Four KBI ports
- Two Internal Op Amps
- Eight MHz bus clocks with FLL
- 1 Internal 12-bit DAC
- Device USB

Features:

- Accurate measurement
- Simple and effective control
- Low cost electronic system
- Analogue amplification internally connected with adjustable reference
- USB connectivity
- Configurable gain amplification and amplificatory configurations
- Capability to probe a lot of different control algorithms and configurations

Chapter 3
System Background

3.1 System Background

A ventilator is a machine designed to mechanically move air in and out of the lungs to intermittently or continuously assist or control pulmonary ventilation. This device is primarily used in intensive therapy to help improve the patients breathing by regulating the flow of gas in the lungs. The most common indicators of the ventilation are the absolute volume and changes of volume of the gas space in the lungs achieved during a few breathing maneuvers. The ventilator is constantly monitored and adjusted to maintain appropriate arterial pH and PaO2.

This system requires a set of sensors for pressure, volume, and flow. The information from the sensors modulates the operations in the MCU. This MCU receives information from the airways, lungs, and chest wall through the sensors, and decides how the ventilator pumps.

The pneumatic system has two air supplies that can be oxygen and air, and can come from a pressurized tank or compressor. Both sources are regulated by two input valves to control mixture composition, which comes from an air tank where the mixture is kept at certain pressure limits. If the mixture composition is correct and is in the right pressure range, the system sends this air to the patient to control breathing. For this system has an input and output valves connected to lungs simulator to control pressure in lungs and the respiratory frequency to maintain patient safety. Figure 3-1 shows the pneumatic system for this ventilator.

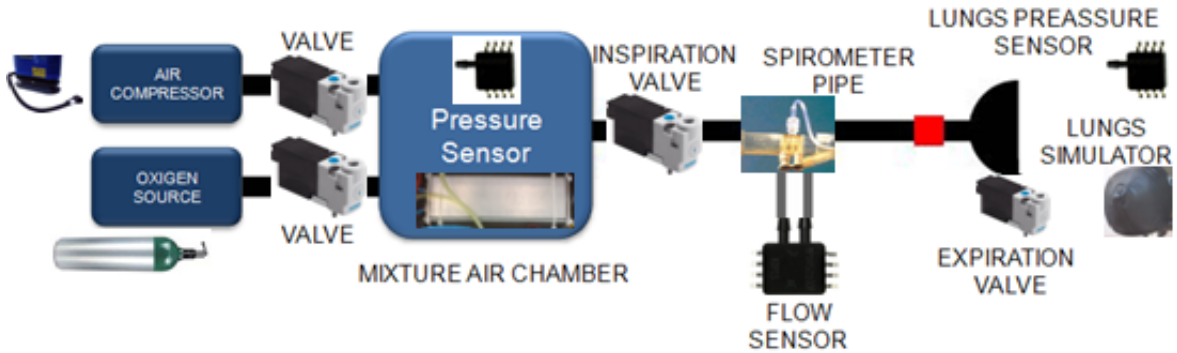

Figure 3-1. Pneumatic system diagram

3.2 System sensors

The signal that shows lung volume is a differential signal, but this is not the signal measured directly from the lungs. To get this signal, it is necessary to transduce the pressure to voltage. This is done by using a pneumotachometer that contains a pressure sensor.

Freescale provides a variety of sensors that use integrated circuits for signal conditioning. This is an advantage because external components are not necessary. However, it is necessary to check the sensor and ADC resolution. If the ADC resolution is greater than the sensor resolution, amplifying the signal is recommended. Some sensors provide differential outputs to pass the signal through an instrument amplifier, when necessary. The sensor that fits volume measurement is a differential pressure sensor that accepts two sources of pressure simultaneously. The output is proportional to the difference of the two sources. It is important to mention that the normal pipeline gas source of a hospital is 50 PSI. This is a measurement that can be taken by Freescale pressure sensors.

3.3 Spirometer

Spirometers measure static pulmonary volumes, except the functional residual capacity (FRC) and total pulmonary capacity (TPC). The measurement is executed after a maximum inspiration that requires the patient to expel the entire volume of air that he or she can. The results are interpreted and compared with the expected values for age, height, sex, and race of the patient. Normal values can fall between 80 to 120 percent of the expected volume; this is due to variations among normal individuals.

Lung volume measurements include:

- Tidal volume (TV)—Amount of gas inspired or expired with each breath (500 ml)

- Inspiratory reserve volume (IRV)—Maximum amount of additional air that can be inspired at the end of a normal inspiration (2500 ml)
- Expiratory reserve volume (ERV)—Maximum volume of additional air that can be expired at the end of a normal expiration (1500 ml)
- Residual volume (RV)—The volume of air remaining in the lungs after a maximum expiration (1500 ml). These measurements can be used in the following equations to express lung capacities:
  - Eqn. 1—Total lung capacity (TLC)
    - TLC=RV+IRV+TV+ERV (6000 ml)
  - Eqn. 2—Vital capacity (VC)
    - VC=IRV+TV+ERV=TLC-RV (4500 ml)
  - Eqn. 3—Functional residual capacity (FRC)
    - FRC=RV+ERV (3000 ml)
  - Eqn.4—Inspiratory capacity (IC)
    - IC=TV+IRV (3000 ml)

3.4 Flow measurement

The Venturi effect is used to measure flow. This is the reduction in fluid pressure that results when a fluid flows through a constricted section of a pipe. This effect is called a jet effect because the velocity of the substance increases on the way from the wide section to the narrow section. The pressure also increases over a smaller surface area; the same force applied to a smaller area equals a higher pressure in that area. According to fluid dynamics, a fluid's velocity must increase as it passes through a constriction to satisfy the conservation of mass, while its pressure must decrease to conserve energy. Equation 5 refers to the Venturi effect

Eqn. 5 Venturi effect :

$$P_1 - P_2 = \frac{p}{2}\left(v_2^2 - v_1^2\right)$$

Where is the density of the fluid, v1 is the (slower) fluid velocity where the pipe is wider, v2 is the (faster) fluid velocity where the pipe is narrower (as seen in the figure). This assumes the flowing fluid (or other substance) is not significantly compressible, even though the pressure varies, the density is assumed to remain approximately constant.

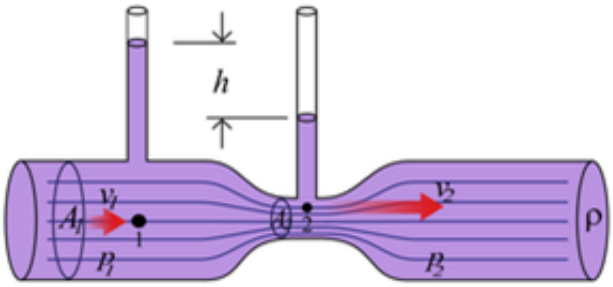

Figure 3-2. Venturi effect graphically demonstration

Figure 3-3 shows Venturi pipes used in industrial and in scientific laboratories for measuring liquid flow. This case system uses a reusable medical Venture pipe, called a D-Lite sensor.

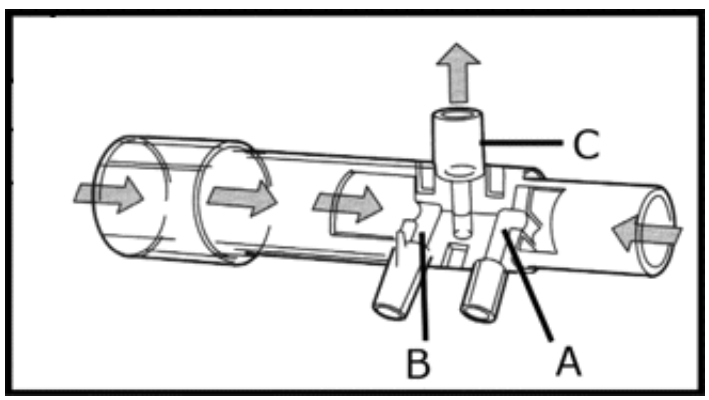

Figure 3-3. Venturi real pipe application for a D-lite sensor

Figure 3-3 shows the sensor with three sampling ports. During inspiration the gas flows from the ventilator to the patient, A measures the total pressure, B measures the static pressure. The difference between the two gives a dynamic pressure, which is proportional to the velocity of gas flow. During expiration the process is reversed. C measures CO2, O2, and anesthetic gas concentration.

A Venturi can be used to measure the volumetric flow rate Q.

Since:

Eqn.6

$$Q = v_1 A_1 = v_2 A_2$$

$$P_1 - P_2 \frac{P}{2}\left(v_2^2 - v_1^2\right)$$

Then:

Eqn. 7

$$Q = A_2\sqrt{\frac{2(P_1-P_2)}{\rho\left(1-\left(\frac{A_s}{A_1}\right)2\right)}} = A_2\sqrt{\frac{2}{\rho\left(1-\left(\frac{A_2}{A_1}\right)2\right)}} * \sqrt{\Delta P}$$

Divide as follows:

Eqn.8

$$Q = A_2\sqrt{\frac{2}{\rho\left(1-\left(\frac{A_2}{A_1}\right)2\right)}} * \sqrt{\Delta P} = K_{sensor} * P_{gas} * \sqrt{\Delta P} = K_{system} * \sqrt{\Delta P}$$

Finally, the pneumatic system has a constant value relating pressure with flow, you can measure directly the differential pressure from the venture, square root it and multiply it for your system constant to obtain the instant volumetric flow.

NOTE

Since differential pressure can be measured, you can experimentally obtain the constant factor of the system by using a flow source.

3.5 Alarm system

An important part of this application is the alarm that indicates different patient parameters such as exhaled volume or airway pressure. The ventilation system must be able to detect whether a breath has been taken. The MCU measures changes in aspiratory flow and pressure by using sensors. If no inspiration is detected within a certain period of time, the monitor sounds an alarm. The conditions programmed depend on each system. PWM cycles can be programmed to sound alarms. Sometimes, the ventilation system uses different alarms for different situations.

3.6 Human machine interface

For the user to have control the system needs an interface with a turn On/Off function, configure, monitor parameters and modes. The system has 20 x 4 characters LCD display that displays the parameters and the four buttons to change them. Figure 3-4 shows the physical interface for the suitcase.

Figure 3-4. System HMI for suitcase

Function description for the Human Machine Interface (HMI):

ON/OFF button — Whenever the button is pressed the system switches the state from OFF to ON or vice versa. To check changes, a red LED on the PCB follows the system states (turn on and off).

NOTE

In case the LED turns on and the system does not react, the reason may be that the system is in interactive mode or that respiration limits are closed.Refer to troubleshooting section.

SELECT button — Is used to modify parameters, when the button is pressed a blinking cursor will turn on and off on the first character of the first line. When the cursor blinks you can modify the parameter in the first line. Not all options are modifiable, measures cannot be modified. The parameter values are updated by the system after you finish the modification mode ,that is when cursor turns off again.

UP/DOWN buttons — When the cursor is not blinking these buttons can be used to explore the cyclic menu, but if the cursor is blinking these buttons modify the parameter values.

MENU — This medical ventilator demo contains an 14 parameter menu; these parameters can be divided into four.

- Measurements — Show measurements for lung pressure, mixing tank pressure, and PCB temperature.

NOTE

These parameters are to be monitored, if select is pressed the cursor blinks and indicates it is in select mode. Up/down buttons will not change the values because it is a real measure.

- Limits and setpoints — Values for limits and setpoints for all the controllable variables, these values have minimum and maximum possible values and are not cyclic.
- Measure units —These modifiable parameters are reflected after each measure and denotes which units are used. Conversions are included and are valid for pressure, and temperature. It also is a cyclic menu.
- Mode of function — This parameter cyclically changes the operational mode of the system, there are three control modes.
  - pressure
  - assisted
  - frequency

  The 14 parameters shown in the menu are:
  - LUNGS_PRESSURE — Pressure measured from the simulating lungs bag
  - TANK_PRESSURE — Pressure retained in the air mixing chamber, this value affects the velocity to fill lungs
  - OPERATION_MODE — Modifies the control mode for the system.
  - TEMPERATURE — Measures the typical highest PCB temperature (near the voltage regulators), this compensates errors in the pressure measurement.
  - OXYGEN_PERCENTAGE — Shows and modifies oxygen percentage from the air mixture.
  - TANK_LOW_PRESSURE — Modifies the tank low limit pressure for all modes.
  - TANK_HIGH_PRESSURE — Modifies the tank high limit pressure for all modes.
  - ALARM_TIME — Sets the alarm time for tank low level pressure, this demonstrates tank air control.
  - LUNGS_LOW_PRESSURE— Modifies the lungs low limit pressure for the pressure control mode.
  - LUNGS_HIGH_PRESSURE — Modifies the lungs high limit pressure for the pressure control mode.
  - INSPIRATION_PERCENTAGE — Modifies the percentage of inspiration in a whole breathing cycle for the frequency mode.
  - RESPIRATION_TIMES — Shows and modifies breaths per minute for the frequency mode.
  - PREASSURE_UNITS — Converts units to PSI, Pascal, H20 cm, Hg mm, and mbar
  - TEMPERATURE_UNITS — Choose between Celsius and Fahrenheit.

3.7 Air and oxygen blender and mix control

The air and oxygen blender provides a precise oxygen concentration by mixing air and oxygen. The concentration can be adjusted to any value from controlled air to 100 percent oxygen. Internally, a proportioning valve mixes the incoming air and oxygen as the oxygen percentage dial is adjusted. A variation in line pressure, flow, or pressure requirements for any attached device will not affect the oxygen concentration.

The preparation of an air and oxygen blender generally consists of attaching a 50 PSI air and oxygen source to the device. After the source gases are attached, inlet pressures may be checked on some blenders by checking the pressure-attached pressure gauge. After the inlet gases are attached and the air and oxygen blender is well secured to a stand or wall mount, it is ready for use.

The MCU uses a PWM to control the blender electro valves through a motor control design. Earlier ventilator designs relied on mechanical blenders to provide premixed gas to a single flow control valve. With the availability of high-quality flow sensors and processing capabilities, accurate mixing becomes possible by using separate flow valves for air and oxygen. Because air already contains about 21 percent oxygen, the total flow control command between the oxygen and air valve is divided ratio metrically. For extreme mix settings, the valve that supplies the minor flow at low total flow requirements may fall below the resolution limits that either flow delivery or measurement can provide. An accurate delivered mix depends on accurate flow delivery, but if accurate and reliable oxygen sensors are used, improved mix accuracy may be possible by feeding back a measured concentration for mix correction. Then, if the patient needs more pressure, the MCU activates the compressor.

This mixture comes into an air tank, where mixture percentages are controlled by pressure with the following formula:

Eqn. 9

$$P_2 = P_1 + (P_3 - P_1) * A\%$$

Where P_1 is the initial gas pressure, P_2 is the pressure that defines the A% (mix percentage) inside the container, P_3 is the final desired and mixed pressure.

The cycle process to control and prepare correct proportions for the mixture are as follows:

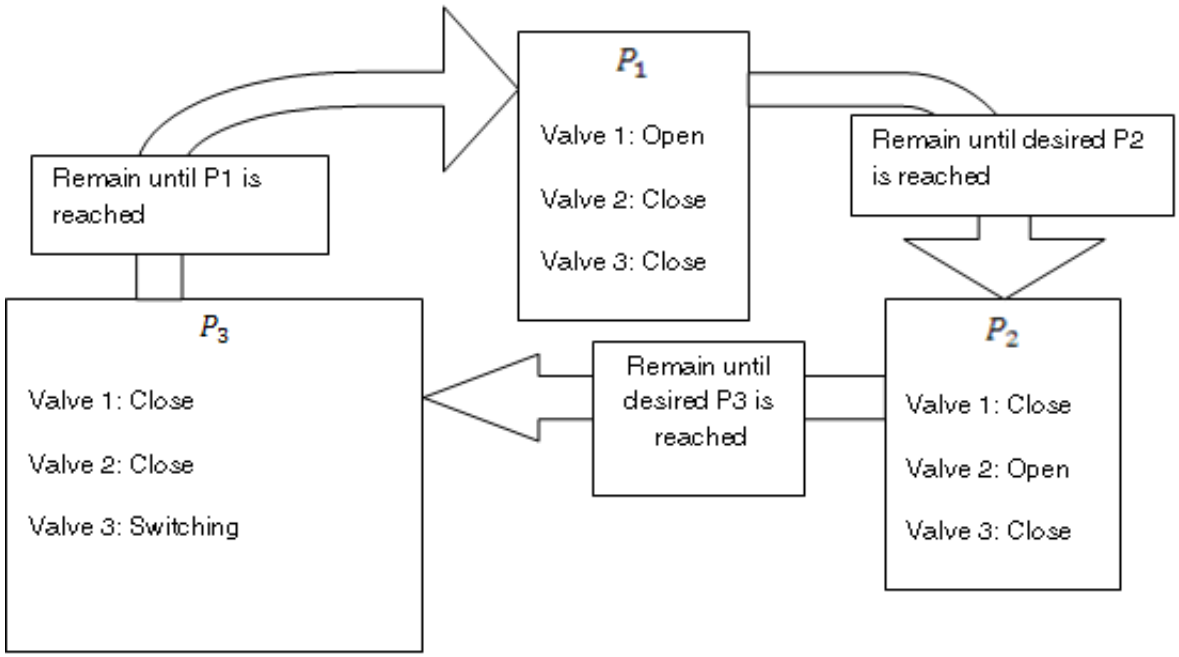

Figure 3-5. Mixture control cycle

3.8 Breathing controller

Ventilators can be divided into five classifications; this depends on how the inhaling process is terminated:

Table 3-1. Ventilator classification

Ventilator types	Description
Pressure–cycled	Inspiration of gas stops when a designated pressure is achieved.
Volume–cycled	Inspiration ends when the desired volume of gas has been introduced
Time–cycled	Inspiration and expiration are programmed the same as the gas flux
Flux–cycled	Inspiration ends when the aspiratory flux is below a pre–determined level
Mixed	The most commonly used; combines the attributes of the other classifications

This medical ventilator can support any of these methods, to show basic functionality this reference design supports only the following modes, but you can probe your algorithms for different modes of operation:

- Pressure Mode—Controlled for certain pressure limits.

- Time–Mode—Respiration and expiration are executed to a certain frequency.
- Assisted—Inspiration starts after a small effort executed by the lung simulator and finishes only one cycle.

Chapter 4
Hardware Design

4.1 System definition

An electronics system for a medical respirator can be complex because of the variety of components and functions that must be accurate.

It must have the following modules:

Power supply—This PCB must power the microcontroller, sensors, display and switch four 200 mA valve, and a 12 A air compressor at different voltages. The requirements of the power supply are 12 A to 12 V (air compressor), 2 A to 5 V (four valves, bright LED, display and differential pressure sensors), and 500 mA to 3.3 V (MCU, buzzer, pressure, and temperature sensors). These modes are on the same PCB. A multi layer PCB will be a requirement. The planes stack selection is an important design action to avoid electrical noise issues.

Communications (USB)—System requires a channel for connectivity with other controlling or monitoring devices to exchange data and provide complete control, for example, anesthetic controllers. USB is a communication channel commonly used for medical devices. This system supports the Freescale's Medical USB stack with a personal healthcare device class (PHDC) that currently supports human interface device (HID), mass storage device (MSD), communications device class (CDC), audio class and PHDC USB.org standard classes. The PHDC enables the software to allow USB connectivity within medical devices by complying with medical industry standards, such as the Continua Health Alliance. Medical ventilators are not supported by PHDC. CDC is a good option for sharing data from these devices.

Signals treatment and measuring from sensors—Reliable measurement is a critical factor for this application. Signal treatment and decoupling are important factors to consider. The system must avoid electrical noise using capacitors and inductors for decoupling. This version does not use external analog treatment, it uses internal op amps to amplify small signals like differential pressure between the simulated lung and ambient without a pumping system. Amplification can be easily modified with software and made

internally. For signals with natural offset (Freescale DP sensors), the reference modification can be an important factor to amplify the signal without cutting it. In this case the Freescale MM microcontroller can be set with a software internal connection of op amp inputs to a Digital to Analog Converter (DAC), changing the reference voltage without having to cut the signal..

Human Interface—This system is able to support some operational modes with different parameter values. It is important to have a user interface for a personalized display or even to explore therapies, events, or for teaching sessions. There is a 20 x 4 character LCD that displays the main parameters and four buttons to explore the 14 option menu. As part of the user interface the system has LED's for alarming, debugging, and a buzzer for alarm events, and therefore the system provides enough information to the user.

Actuators controlling—System requires controlling several electromechanical actuators that demand high current consumption and generates electrical noise to the system by EMI. The system requires a good electrical design to avoid noise generation and device protection like optical switch and snubbers for switching. It is also critical to support and provide enough current. The system uses Darllington transistors, optoislolators, and relays to provide correct power management.

Figure 4-1 is the medical ventilator block diagram.

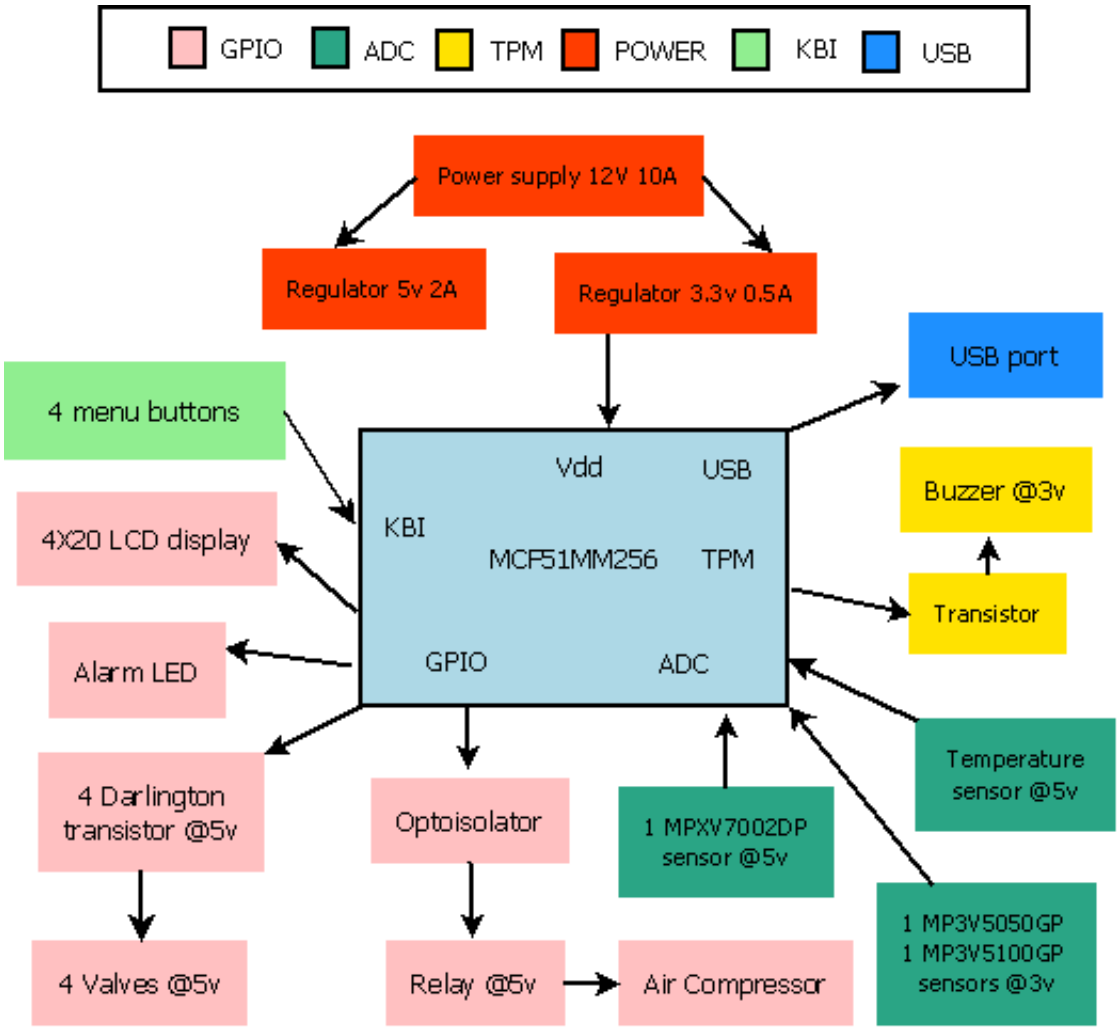

Figure 4-1. Electronic system block diagram

4.1.1 Microcontroller

One of the more critical modules for this application is signal treatment and measuring, this is because the respirator acts according to the data acquired. If this data is not reliable, the ventilator may do incorrect operations that can risk the patient's health. It is important to select the appropriate MCU and sensors for instrumentation applications. Freescale offers reliable MCUs for medical instrumentation. Two examples are the MC9S08MM (8-bit MCU) and the MCF51MM (32-bit MCU). For this reference design the MCU chosen was the MCF51MM256.

The MCF51MM256 provides ultra-low-power operation, USB connectivity, graphic display support, and unparalleled measurement accuracy in a single 32-bit microcontroller. This allows device designers to create more fully featured products at a lower cost.

The MCF51MM256 is ideal for medical applications or any other application requiring a significant amount of precision analog such as instrumentation and industrial control.

The MCF51MM256 is part of the Freescale Flexis™ microcontroller series. This is a summary of the MCF51MM256 features.

- Up to 50.33 MHz ColdFire V1 core speed and 25 MHz bus speed
- 256 K flash
- 32 K SRAM
- Two ultra-low-power stop modes
- Time of the Day (TOD)
- Two flexible operational amplifiers (OPAMP)
- Two trans-impedance amplifiers (TRIAMP)
- Multi-purpose clock generator (MCG)
- Dual-role full-speed USB On-The-Go (OTG) controller and transceiver
- Two serial communications interfaces (SCI)
- Two serial peripheral interfaces (SPI)
- Analog comparators
- 16-bit Analog-to-Digital converter (ADC)
- 12-bit Digital-to-Analog converter (DAC)
- Mini-FlexBus
- Analog comparator with selectable interrupt (PRACMP)
- I2C interface
- Programmable delay block (PDB)
- Carrier modulation timer (CMT)
- Two timer modules (TPM)
- Voltage Reference Output (VREFO)
- Up to 68 GPIOs and 16 rapid GPIOs

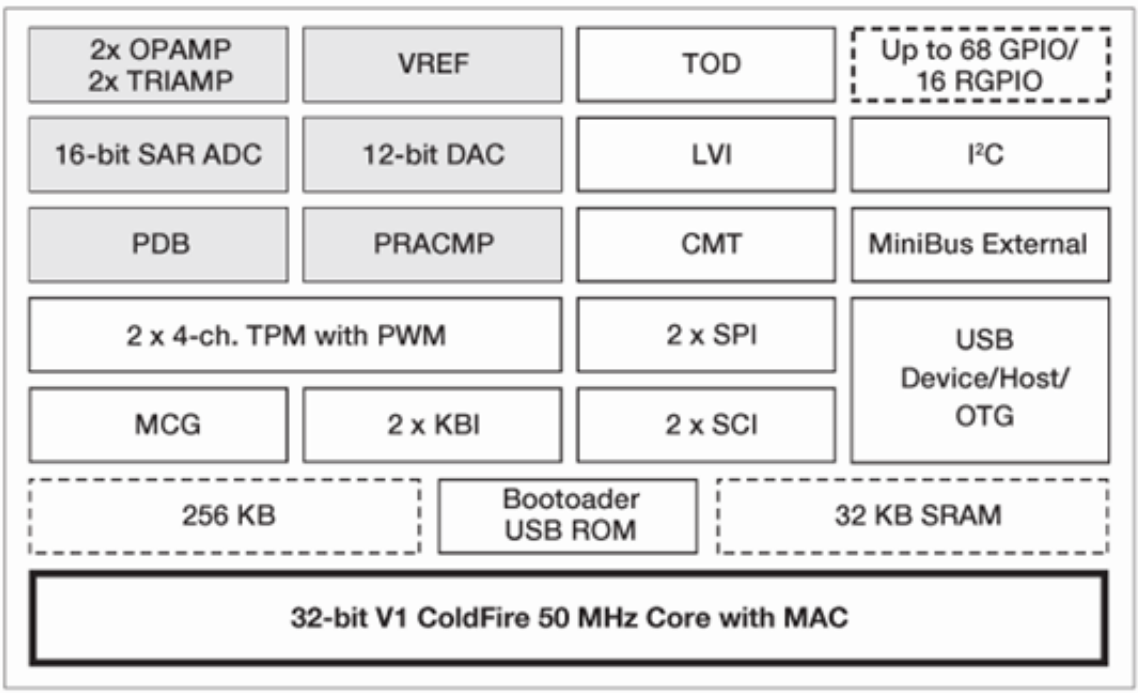

Figure 4-2. MCF51MM Block diagram

4.1.2 Pressure sensors

To measure accurately the control system, it is important to have the correct sensors. The following pressure sensors are used:

MPXv5050GP—0 to 50 kPa integrated silicon pressure sensor, temperature compensated, and calibrated

Key features:

- 2.5 % maximum error over 0° to +85°C
- Ideally suited for microprocessor or microcontroller-based systems
- Temperature compensated over -40°C to +125°C
- One analog output voltage (0 - 3.3 V), CASE 1369-01.

Figure 4-3 shows voltage versus the pressure graph.

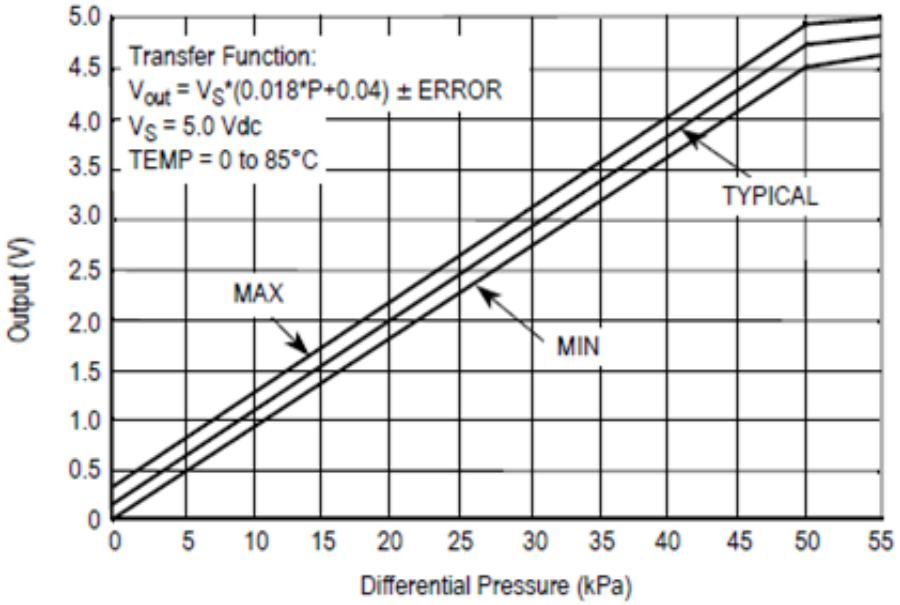

Figure 4-3. MPXV5050GP output vs. pressure signal waveform

MPXv5100GP— 0 to 100 kPa integrated silicon pressure sensor, temperature compensated, and calibrated

Key features:

- 2.5% maximum error over 0° to +85°C
- Ideally suited for microprocessor or microcontroller-based systems
- Temperature compensated over -40°C to +125°C
- One analog output voltage (0 - 3.3 V), CASE 1369-01

Figure 4-4 shows the voltage versus the pressure graph.

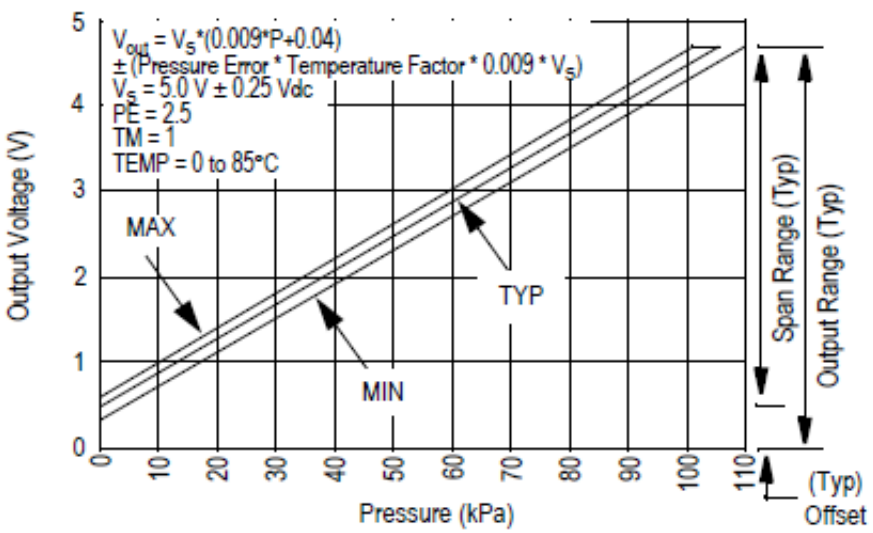

Figure 4-4. MPXV5100GP output vs. pressure signal waveform

MPXV7002DP— –2 to 0 and 0 to 2 kPa integrated silicon differential pressure sensor, temperature compensated, and calibrated

Key features

- .0% maximum error over 0° to 85°C
- Temperature compensated over 10° to 60°C
- Available in differential and gauge configurations
- One analog output voltage (0 - 5 V), CASE 1351-01

Figure 4-5 shows the voltage versus the pressure graph.

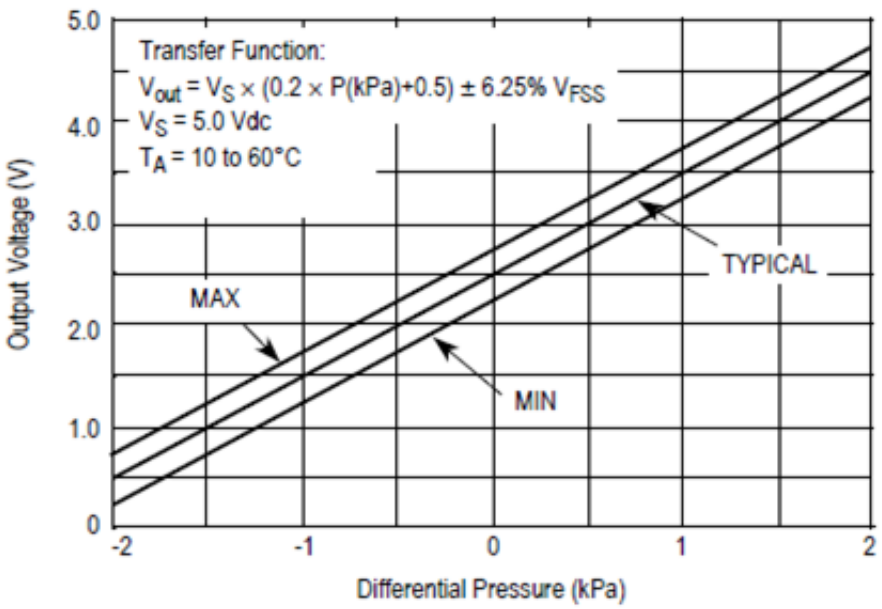

Figure 4-5. MPXV7002DP output vs. pressure signal waveform

The MPX series piezoresistive transducer is a state-of-the-art monolithic silicon pressure sensor designed for a wide range of applications, mainly for those using a microcontroller or microprocessor with A/D inputs. These patented, single element transducers combine advanced micromachining techniques, thin-film metallization, and bipolar processing to provide an accurate high level analog output signal proportional to the applied pressure.

4.1.3 Power supply

The entire ventilator system is powered by a 12 V 15 A power supply connected to a 120/220 V AC power line that may be switched according to the power supply. This power supply powers the 12 V 10 A air compressor with the 5 V coil relay. The system also will have the L78S05CV 5 V regulator at 2 A used to drive four 200 mA electro

valves, a temperature sensor, switch the 5 V coil of relay, and a differential pressure (flow) sensor. And finally, a TLV2217-33KCSE3 a 3 V at 500 mA regulator to power the MCF51MM MCU, an 80mA buzzer, and the pressure sensor.

Table 4-1. Voltage supply requirements

Device	Voltage	Notes
MCF51MM	2.4 V – 3.8 V	At frequency between 20 and 50 MHz
	1.8 V – 3.8 V	At frequency less than 20 MHz
MP3V5050	2.7 – 3.3 V	
MPXV7025	4.75 V – 5.25 V	
Air compressor	12 V DC	10 A consumption
Electro valves	4.5 V – 5.5 V	200 mA consumption
Buzzer	4.75 V – 5.25 V	80 mA consumption

Proper decoupling and a correct pad stack election is important to support all this power without electrical noise and electromagnetic interference (EMI) on the printed circuit board (PCB). The power supply also contains the BYW29E-150,127 a 30 V 2 A diode that protects the electrical circuit of inverse currents and the SMCJ13A a diode TVS to 13 V 1500 W, that protects for transients.

Considering the signal planes needed (12 V, 5 V, 3.3 V and 3.3 V for MCU), your pad stack selection can be with inner power signal levels and outside ground levels, acting like a faraday gauge and reducing EMI.

Figure 4-6 shows the pad stack selection in the layout tool for a standard board thickness of 62 mils.

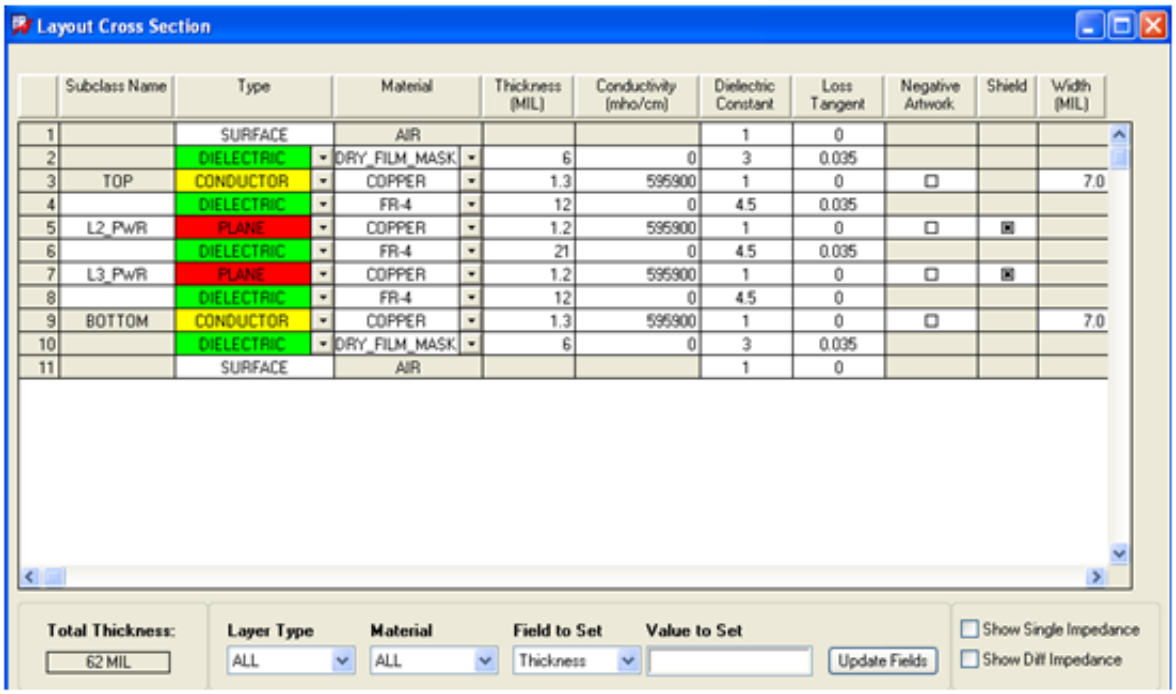

Figure 4-6. Pad stack selection

4.1.4 Analog To Digital Converter (ADC)

To measure sensor analog signals for pressure calculating, the system needs an accurate ADC. The MM MCU has 24 ports with 16-bit ADC that can be used to read different sensors. Another useful feature for this MCU is the Op Amp output that can be interconnected via software to the ADC channel. This is useful for simplifying routing or for prototyping phases.

In this case, two pressure sensors and one temperature sensor are connected directly to the ADC, while the signal from two differential pressure sensors are connected from the outputs of the Op Amps.

Before signals go to the ADC, it is necessary to amplify them or improve them, here the internal General Purpose Op Amps (GPAMP) is a useful module that is able to probe configurations, gain values, and other input and output processes. In this case the output of the Digital to Analog converted is internally connected as an input of the GPAMP to improve the signal amplification. DAC can be used to change the reference voltage and prevent cutting signals.

4.1.5 Keyboard Interruptions (KBI)

The use of KBIs for the interface buttons eliminates the necessity for extra components due to the internal pull-ups, and simplifying code by directly turning on interrupts. One KBI port per button is needed.

4.1.6 Universal Serial Bus (USB)

USB is a commonly used communication channel for medical applications. Medical standards are moving to create its own USB class and subclasses (PHDC) to create major medical device environments that can offer the user better health care services.

4.1.7 General Purpose Input Outputs (GPIO)

The system needs GPIOs to control the LCD display, switch valves, and the compressor. The MCU has the Drive Strength feature to provide more current when needed, for example for the actuators.

The GPIO pulses must reflect the MCU control decisions. To control electrical and electromechanic actuators the power stage must have enough current requirements avoiding electrical noise generation trying to reduce and prevent components damage risks

4.2 Bill Of Materials (BOM)

A list of all the components for this demo are shown in the table below.

Table 4-2. Bill of materials

Item #	Reference Designator	Manufacturer part #	Manufacturer's Name	Description	QTY	RoHS Compliant (Yes or No)
PCBA BOM						
1	BH1,BH2,BH3,BH4	MOUNTING HOLE	NA	NA	4	NA
2	C1,C13,C14	UWT1V470MCL1GS	Nichicon	CAP 47UF 35V ELECT WT SMD	3	Yes
3	C2,C3,C5,C6,C7,C8, C10,(C11DNP), (C12DNP),C15,C16,C17,C20,C21,C28, C29,C30,C31,C32	GRM21BR71H104KA01L	Murata Electronics	CAP CER .1UF 50V 10% X7R 080	17	Yes

Table continues on the next page...

Table 4-2. Bill of materials (continued)

Item #	Reference Designator	Manufacturer part #	Manufacturer's Name	Description	QTY	RoHS Compliant (Yes or No)
4	C4	EEE-1EA100SR	Panasonic - ECG	CAP ELECT 10UF 25V VS SMD	1	Yes
5	C9	GRM216R71H103 KA01D	Murata Electronics	CAP CER 10000PF 50V 10% X7R 0805	2	Yes
6	C19,C34	EEE-1HA4R7AR	Panasonic - SSG	CAP ELECT 4.7UF 50V VS SMD	2	Yes
7	C22,C23,C24,C25, C26,C27	DNP	DNP	DNP	0	DNP
8	C35	GRM21BR71H474 KA88L	Murata Electronics	CAP CER .47UF 50V X7R 0805	1	Yes
8	D18	BYW29E-150,127	NXP	DIODE RECT UFAST 150V TO220AC	1	Yes
9	D2,D8,D9,D10,D11 ,D12,D13	B230A-13-F	Diodes Inc	DIODE SCHOTTKY 30V 2A SMA	6	Yes
10	D14	LNJ406K5YUX	Panasonic - SSG	LED AMBER S-J TYPE 0805	1	Yes
11	D15	LNJ306G5UUX	Panasonic - SSG	LED GREEN S-J TYPE 0805	1	Yes
12	D16	LNJ806K5SRX	Panasonic - SSG	LED SOFT ORANGE S-J TYPE 0805	1	Yes
13	D17	TLHK5800	Vishay/ Semiconductors	LED 5MM RED 1000MCD ALINGAP GAAS	1	Yes
14	D19	SMCJ13A	Littelfuse Inc	DIODE TVS 13V 1500W UNI 5% SMC	1	Yes
15	J1,J2,J3,J4,J5,J6	1725656	Phoenix Contact	CONN TERM BLOCK 2.54MM 2POS	6	Yes
16	J12	1-640456-6	Tyco Electronics	LCD CONN HEADER VERT 16POS .100 TIN	1	Yes
17	J13	87227-3	Tyco Electronics	CONN HEADER VERT .100 6POS 15AU	1	Yes
18	J14,J15	Jumper 3Pin	3-644456-3	CONN HEADER VERT 3POS .100 TIN	2	Yes
19	J16	UX60A-MB-5ST	Hirose Electric Co Ltd	CONN RECEPT MINI USB2.0 5POS	1	Yes

Table continues on the next page...

Table 4-2. Bill of materials (continued)

Item #	Reference Designator	Manufacturer part #	Manufacturer's Name	Description	QTY	RoHS Compliant (Yes or No)
20	LS2	CSQG703BP	CUI Inc	BUZZER MAG 2.0KHZ 3V 12MM	1	Yes
21	L1,L2,L3	BLM21PG221SN1D	Murata	FERRITE CHIP 220 OHM 2000MA 0805	3	Yes
22	L4,L5	BKP1608HS271-T	Taiyo Yuden	FERRITE BEAD 270 OHM 0603	2	Yes
23	Q1,Q2,Q3,Q4,Q5	IP31C	Fairchild Semiconductor	TRANS NPN EPITAX 100V 3A TO-220	5	Yes
24	R1	ERJ-6GEYJ472V	Panasonic - ECG	RES 4.7K OHM 1/8W 5% 0805 SMD	1	Yes
25	R2,R9	ERJ-6GEYJ331V	Panasonic - ECG	RES 330 OHM 1/8W 5% 0805 SMD	2	Yes
26	R3	ERJ-6GEYJ105V	Panasonic - ECG	RES 1.0M OHM 1/8W 5% 0805 SMD	0	Yes
27	R4,R5,R6,R7,R12	ERJ-6ENF4700V	Panasonic - ECG	RES 470 OHM 1/8W 1% 0805 SMD	5	Yes
28	R8	ERJ-6GEYJ181V	Panasonic - ECG	RES 180 OHM 1/8W 5% 0805 SMD	1	Yes
29	R10,R11	ERJ-6GEYJ221V	Panasonic - ECG	RES 220 OHM 1/8W 5% 0805 SMD	2	Yes
30	R13	ERJ-6GEYJ103V	Panasonic - ECG	RES 10K OHM 1/8W 5% 0805 SMD	1	Yes
31	R14	3352W-1-103LF	Bourns Inc.	POT 10K OHM THUMBWHEEL CERM ST	1	Yes
32	R15,R23	ERJ-6ENF1693V	Panasonic - ECG	RES 169K OHM 1/8W 1% 0805 SMD	2	Yes
33	R16,R24	ERJ-6GEYJ334V	Panasonic - ECG	RES 330K OHM 1/8W 5% 0805 SMD	2	Yes
34	R17,R18,R19,R20, R21,R22	ERJ-6GEY0R00V	Panasonic - ECG	RES 0.0 OHM 1/8W 0805 SMD	6	Yes
35	R25,R26	RMCF0805JT33R0	Stackpole Electronics Inc	RES 33 OHM 1/8W 5% 0805 SMD	2	Yes

Table continues on the next page...

Table 4-2. Bill of materials (continued)

Item #	Reference Designator	Manufacturer part #	Manufacturer's Name	Description	QTY	RoHS Compliant (Yes or No)
36	SW1	KSR221GLFS	C&K Components	SWITCH TACT SILVR 200GF GULLWING	1	Yes
37	SW2,SW3,SW4,SW5,D16	640456-2	Tyco Electronics	IBUTTON TACTILE PROBE CONN HEADER VERT 2POS .100 TIN	5	Yes
38	TP1	T POINT TH	NA	NA	0	NA
39	U2	MCF51MM256CLL	Freescale Semiconductor	32 bits Medical MCU	1	Yes
40	U5	L78S05CV	STMicroelectronics	IC REG POSITIVE 2A 5V TO-220	1	Yes
41	U6	TLV2217-33KCSE3	Texas Instruments	IC LDO REG FXD-VOLT 3.3V TO-220	1	Yes
42	U15	4N27M	Fairchild Semiconductor	OPTOCOUPLER TRANS-OUT 6-DIP	1	Yes
43	U10	MPXV5100GP	Freescale Semiconductor	IC PRESSURE SENSOR 8-SOP 14PSI	1	Yes
44	U12	MP3V5050GP	Freescale Semiconductor	IC PRESSURE SENSOR 8-SOP 7PSI	1	Yes
45	U11,U16	MPXV7025DP	Freescale Semiconductor	PRESSURE SENSOR DUAL PORT 8-SOP	2	Yes
46	U13	LM94021BIMG/NOPB	National Semiconductor	IC TEMP SENSOR MULTI-GAIN SC70-5	1	Yes
47	U14	JSM1A-12V-5	Panasonic Electric Works	RELAY AUTO 15A 12VDC SEALED PCB	1	Yes
48	Y2	NX3225SA-16.000000MHZ	NDK	CRYSTAL 16.000000 MHZ 8PF SMD	0	Yes
System BOM						
49	NA	MHP1-AS-2-M3	FESTO	2/2 Electrovalves plate	2	Yes
50	NA	MHP1-M4H-2/2G-M3-TC	FESTO	2/2 Miniaturized Electrovalves	2	Yes
51	NA	MHP1-AS-3-M3	FESTO	3/2 Electrovalves plate	2	Yes
52	NA	MHP1-M4H-3/2G-M3-TC	FESTO	3/2 Miniaturized Electrovalves	2	Yes

Table continues on the next page...

Table 4-2. Bill of materials (continued)

Item #	Reference Designator	Manufacturer part #	Manufacturer's Name	Description	QTY	RoHS Compliant (Yes or No)
53	NA	QSM-M3-4	FESTO	Fittings	8	Yes
54	NA	KMH-0,5	FESTO	Electrical Valves connectors	4	Yes
55	NA	QSY-8-4	FESTO	Y reduction connection	1	Yes
56	NA	QS-1/4-4	FESTO	Reduction 1/4 to 4 mm	4	Yes
57	NA	4mm Hose	FESTO	4mm air hoses	2	Yes
58	NA	110B 6 2	NA	3/4 to 3/8 reduction	2	Yes
59	NA	110B 12 6	NA	3/8 to 5/32 reduction with fast fitting	2	Yes
60	NA	D703.19	NA	3/4 CPVC femal adapter	2	Yes
61	NA	NA	NA	Pressure sampling line	1	Yes
62	NA		GE Health	D-Lite sensor (Venturi)	1	Yes
63	NA		GE Health	Espirometry hoses	1	Yes
64	NA	NA	NA	Color buttons	4	Yes
65	NA		Acteck	Computer Power Supply with power cord	1	Yes
66	NA		Pelican	Pelican rainstorm suitcase	1	Yes
67	NA		Dogma Studio	Aluminum panel with design	1	Yes
68	NA	NA	Sunmed	Anesthetic bag	1	Yes
69	NA	NA	NA	Documentation CD	1	Yes
70	NA	NA	NA	Demo one pager document	1	Yes
71	NA	NA	Hudson	Anesthetic patient circuit	1	Yes
72	NA	5-87499-8	Tyco Electronics	CONN HOUSING 30POS .100 SINGLE	1	Yes
73	NA	NA	NA	AW 24 Wire	1	Yes
74	NA	5-103171-5	Tyco Electronics	CONN SOCKET 22-26AWG 30AU CRIMP	30	Yes
75	NA	I300	Goodyear	Air compressor	1	Yes

Table continues on the next page...

Table 4-2. Bill of materials (continued)

Item #	Reference Designator	Manufacturer part #	Manufacturer's Name	Description	QTY	RoHS Compliant (Yes or No)
76	NA	1375820-2	Tyco Electronics	CONN RCPT HSNG 2POS CST-100 II - I-button	1	Yes
77	NA	4-643814-6	Tyco Electronics	CONN RECEPT 16POS 24AWG MTA100 - Display	1	Yes
78	NA	C-51847NFJ-SLW-ADN	Optrex America Inc	LCD MOD CHAR 20X4 WHT TRANSFLECT	1	Yes
79	NA	S24453	Keystone Electronics	STANDOFF HEX M4 THR ALUM 10MM	4	Yes
80	NA	NA	IDEAS	Air chamber	1	Yes

4.3 Schematics

In this section electrical connections are presented by schematic blocks.

4.3.1 MCU block

The following figure shows the MCU connection tags.

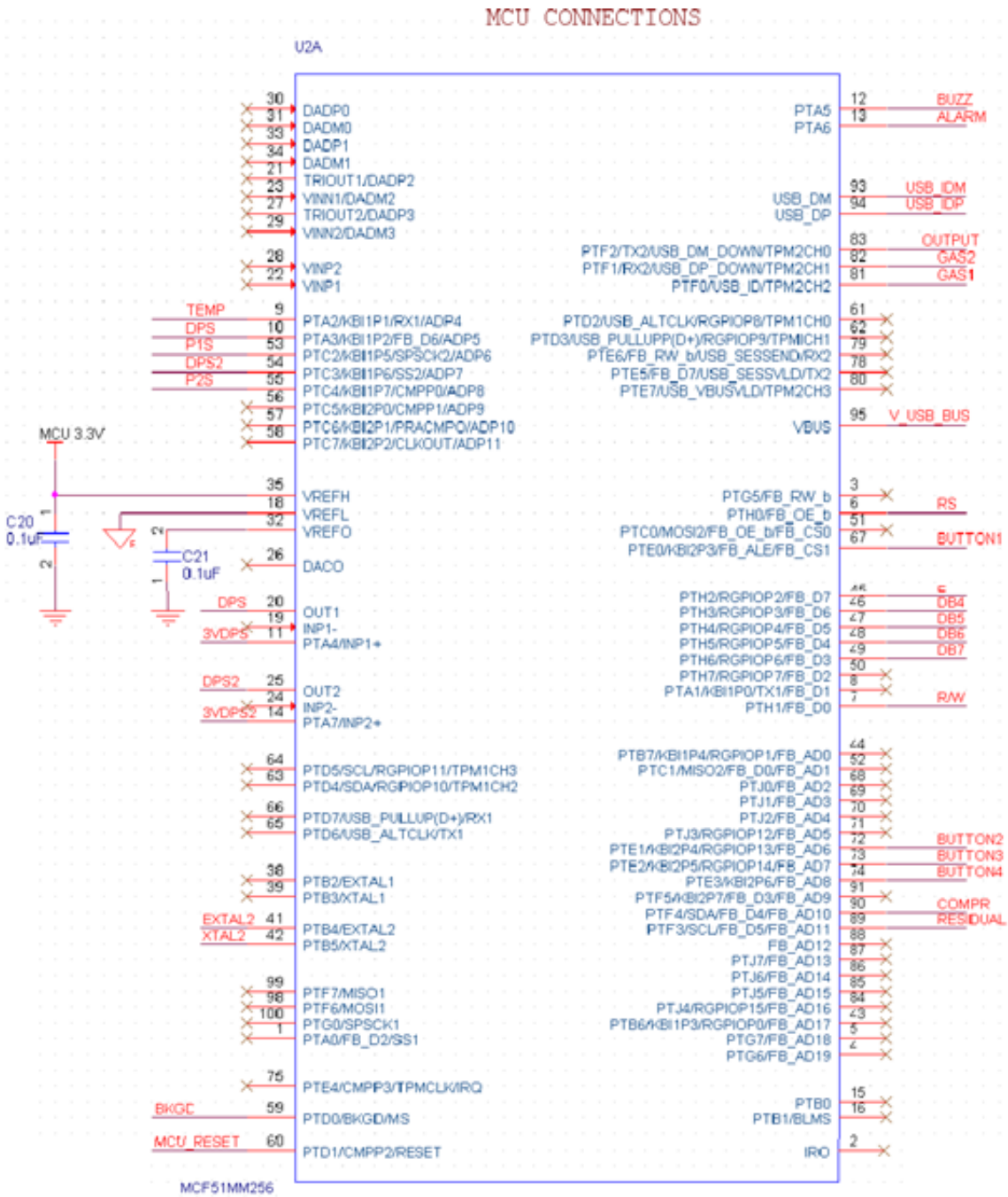

Figure 4-7. MCU

4.3.2 Power supply

Power supply connections will have some decoupling capacitors, separated 3.3 V for the microcontroller, and a separated ground for analog ground. It is protected against inverse current and transients with a diode and a TVS diode. It is important have test points and

LEDs for debugging the power supply. Be sure of current capabilities for voltage regulators, to support the entire system. Figure 4-8 shows the actual power supply schematic diagram.

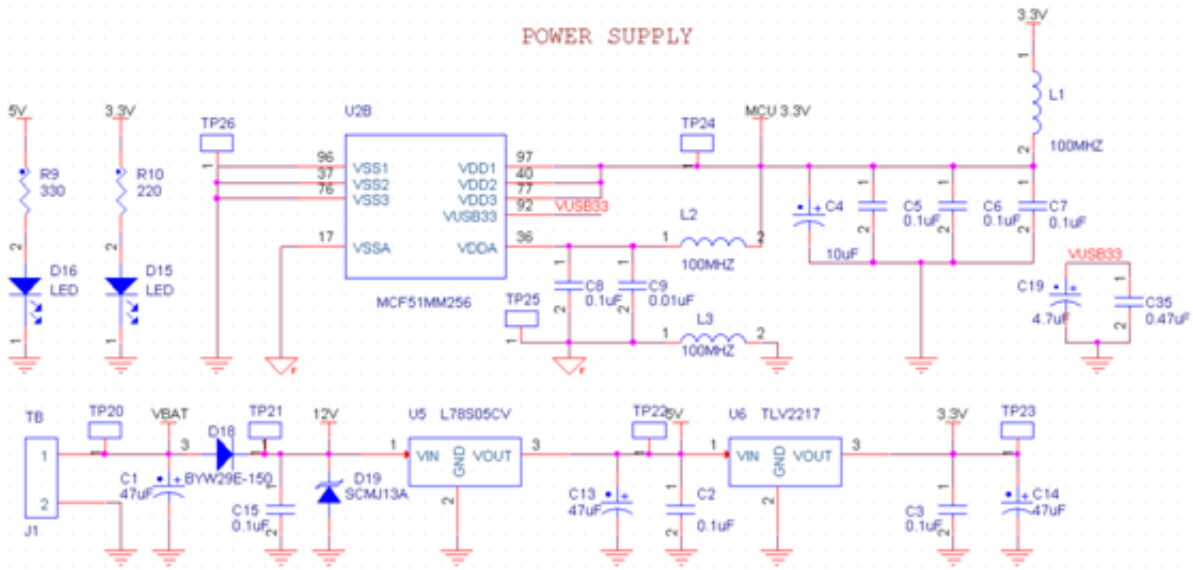

Figure 4-8. Power supply

4.3.3 BDM and Reset

Background Debug Mode (BDM) and reset are fundamental signals for an appropriate circuit using Freescale microcontrollers. The BDM is needed to re-flash program memory and continue with system developing. And reset helps the user in case of some issues like exceptions or like a fast restarting program in an emergency case.

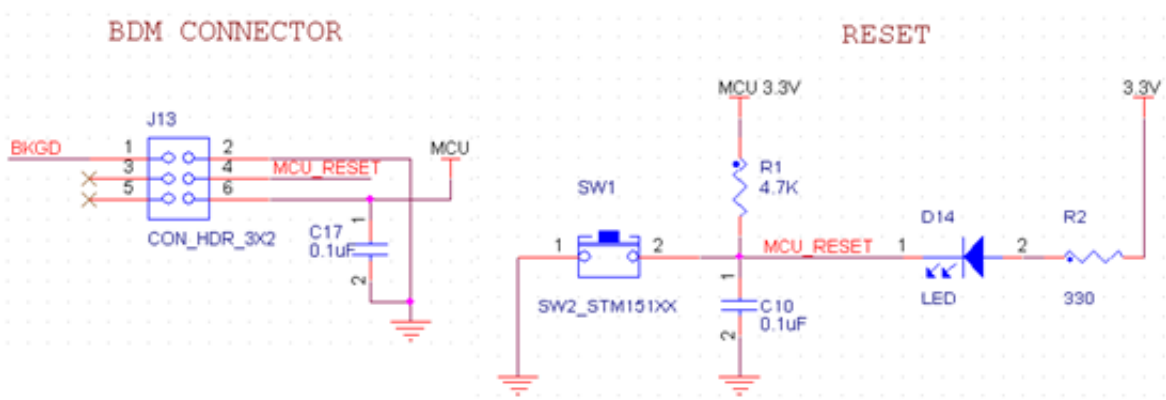

Figure 4-9. BDM and reset

4.3.4 USB and Clock

The figure below shows USB connections for device mode. It is designed to eliminate electrical noise, for layout it is important that the USB data routes have similar distances and a close distance because they are differential signals. These constraints are helpful for signal integrity. For a correct USB function, the MCU must have an external clock with a recommended frequency of 16 MHZ, if this external clock is not used the USB communication can have synchronization issues and frequently disconnect.

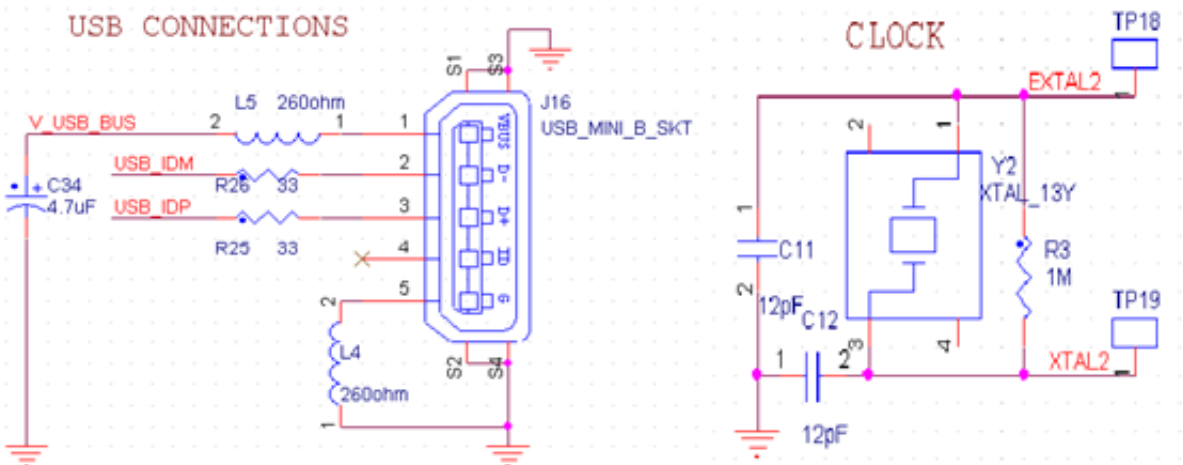

Figure 4-10. USB and clock

4.3.5 Actuators switching

Power stages for actuator switching are shown in figure 16. For the air compressor MCU first switch a transistor based optoisolator (4N27M) to separate grounds. Then switch the relay coil with 5 V, this coil needs 100 mA to switch, the relay then provides 15 A at 12 V for the air compressor.

About valves, MCU ports switch the base of Darlington transistor (TIP31C 500mA) to switch the 5 V at1 W valves, it means 200 mA.

A resistor to the MCU is needed for both cases to control the MCU pin current consumption. The use of RCD snubbers protects inverse currents.

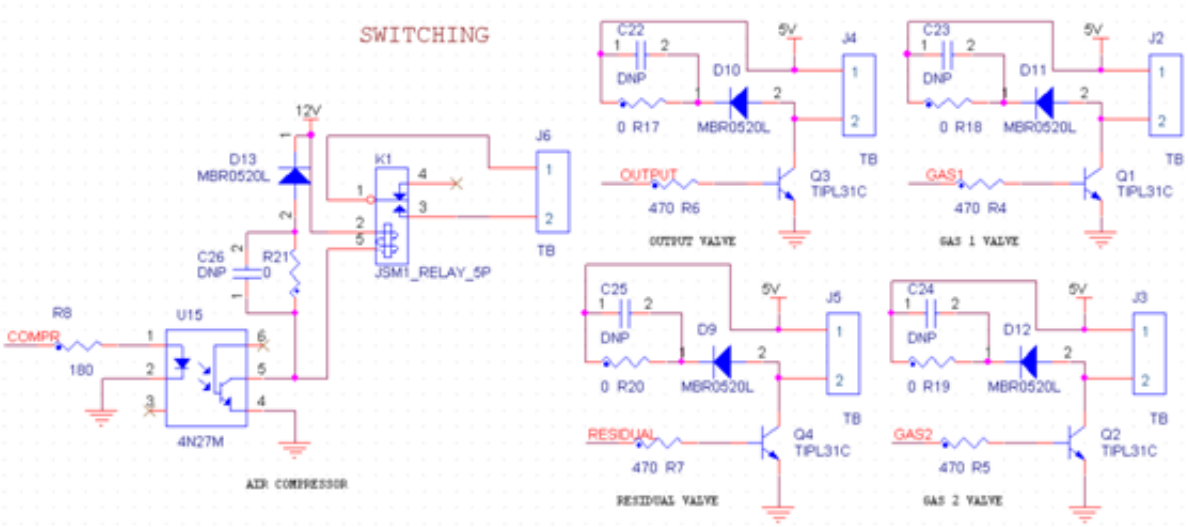

Figure 4-11. Actuators switching

4.3.6 Alarms

This system has two types of alarms for the patient, visual, and audio. Visual is a bright LED, and audio is the buzzer, which uses the same switching circuit as valves to ensure correct working. This device can be switched with the PWM or IO with timers.

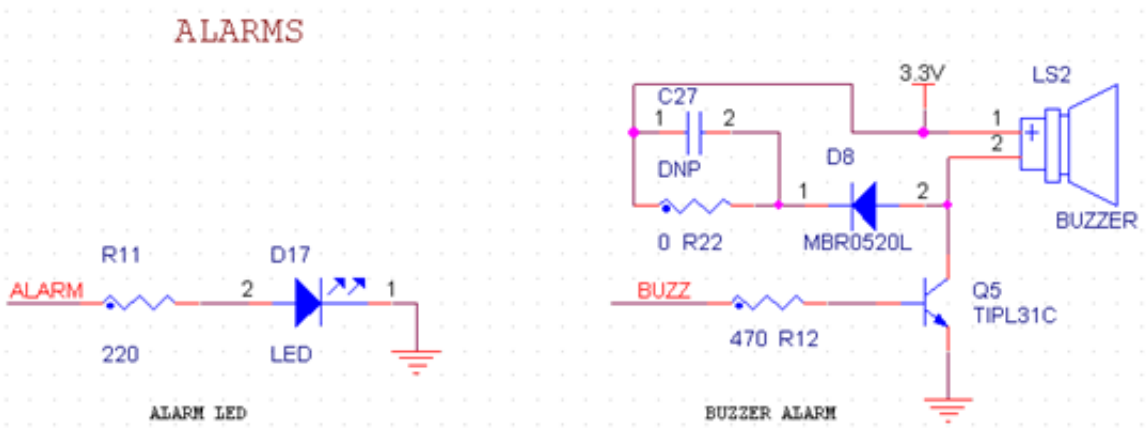

Figure 4-12. Alarms

4.3.7 Sensors

Figure 18, shows single decoupled sensor diagram connections. Its output is connected with internal Op Amps for amplification and isolation of current consumption after voltage dividers. For the Temperature sensor, there are two jumpers for gain selection. This configuration must be changed in software macros for future versions to connect gain selection signals to the MCU input pins for automatic adaptation.

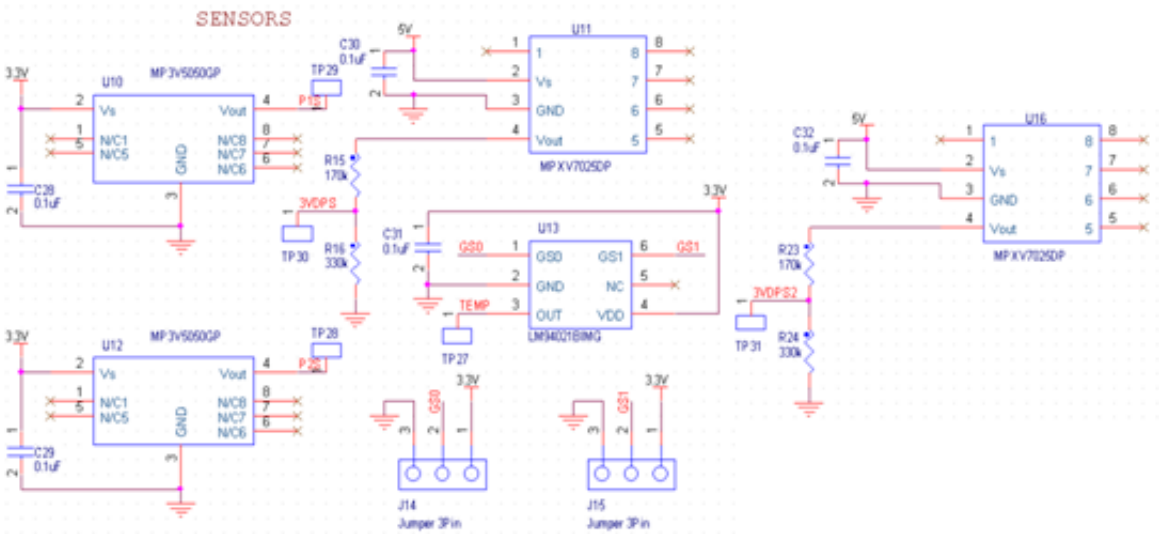

Figure 4-13. Sensors

4.3.8 Human Machine Inteface (HMI)

The human machine interface system has four buttons and a 20 x 4 characters LCD. For button connections there are no external pull-ups, they can be enabled by the software. The LCD has a four parallel channel configuration. A potentiometer is used to change the character darkness.

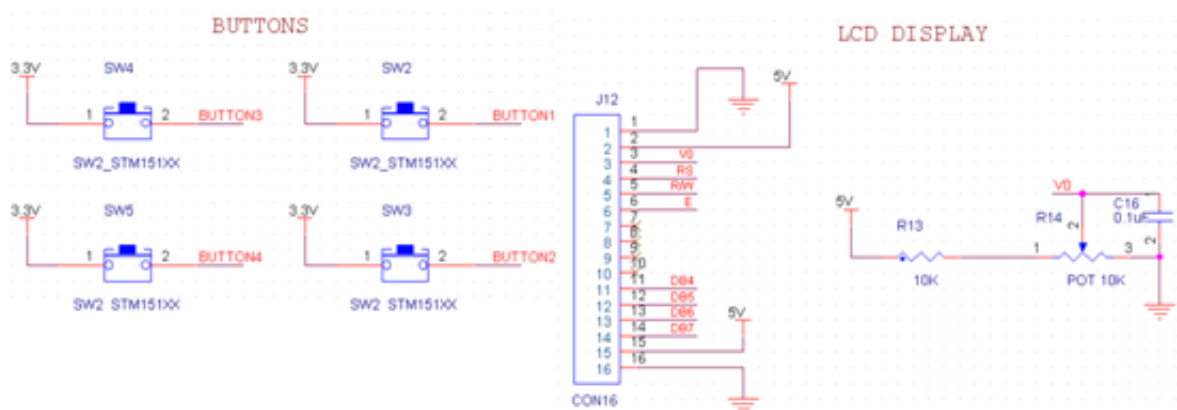

Figure 4-14. Human Machine Interface

4.4 Layout

Correct PCB layout maintains signal integration, see the section (3.1.3) Power Supply.

The pad stack chosen:

- Top—Ground plane and routing 1.
- Inner 1—3.3 V and MCU 3.3 Planes, and air compressor switching plane 1.
- Inner 2—5 V and air compressor switching plane 2.
- Bottom—Ground plane and routing 2.

4.4.1 Layout design

The layout design created on the Allegro PCB editor program is shown in the following figures.

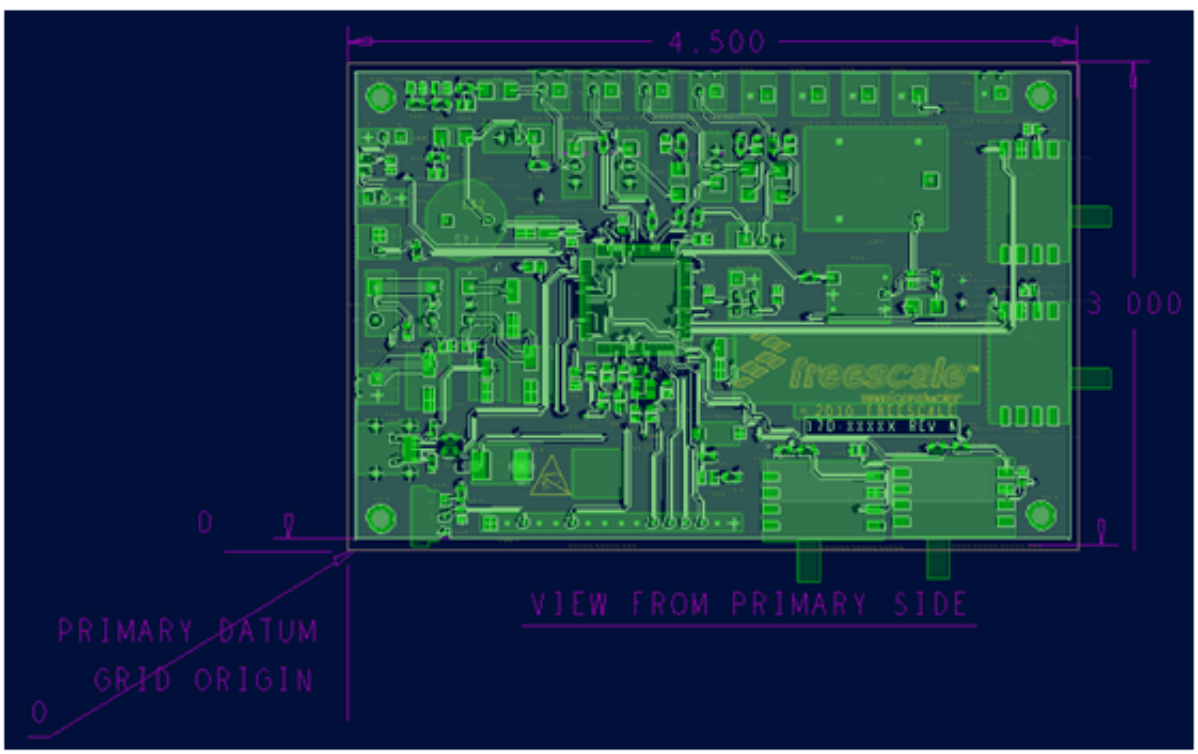

Figure 4-15. Ventilator layout design (top layer—routing and ground plane)

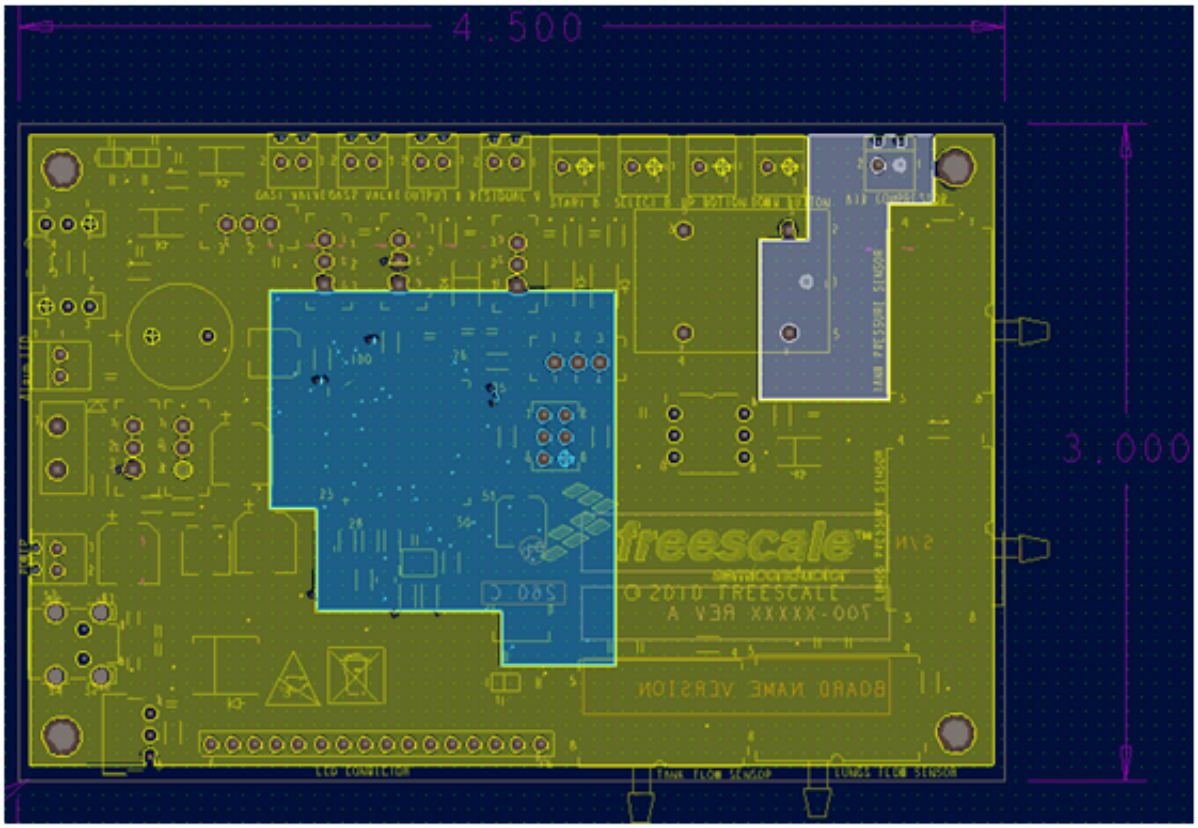

Figure 4-16. Ventilator layout design (Power layer 1—3.3 V and the MCU 3.3 V planes and one 12 V switching route)

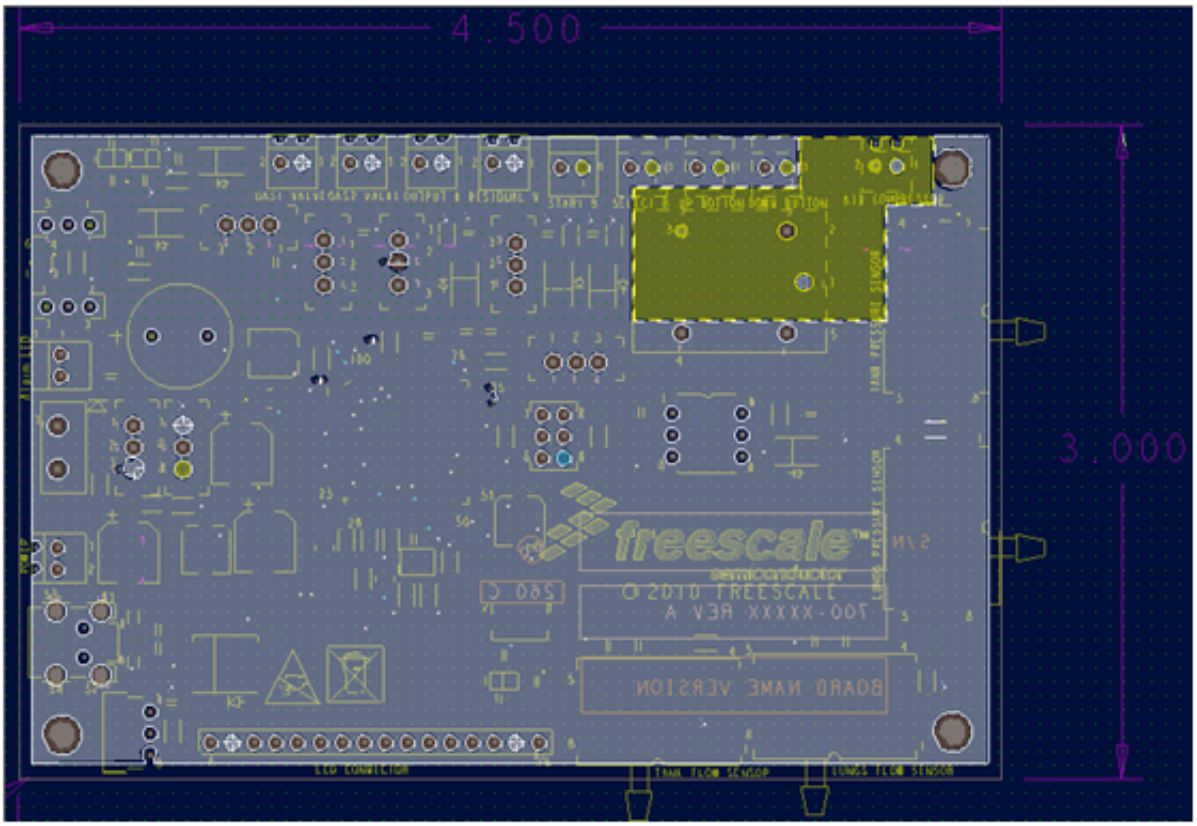

Figure 4-17. Ventilator layout design (Power layer 2—5 V plane and one 12 V switching route)

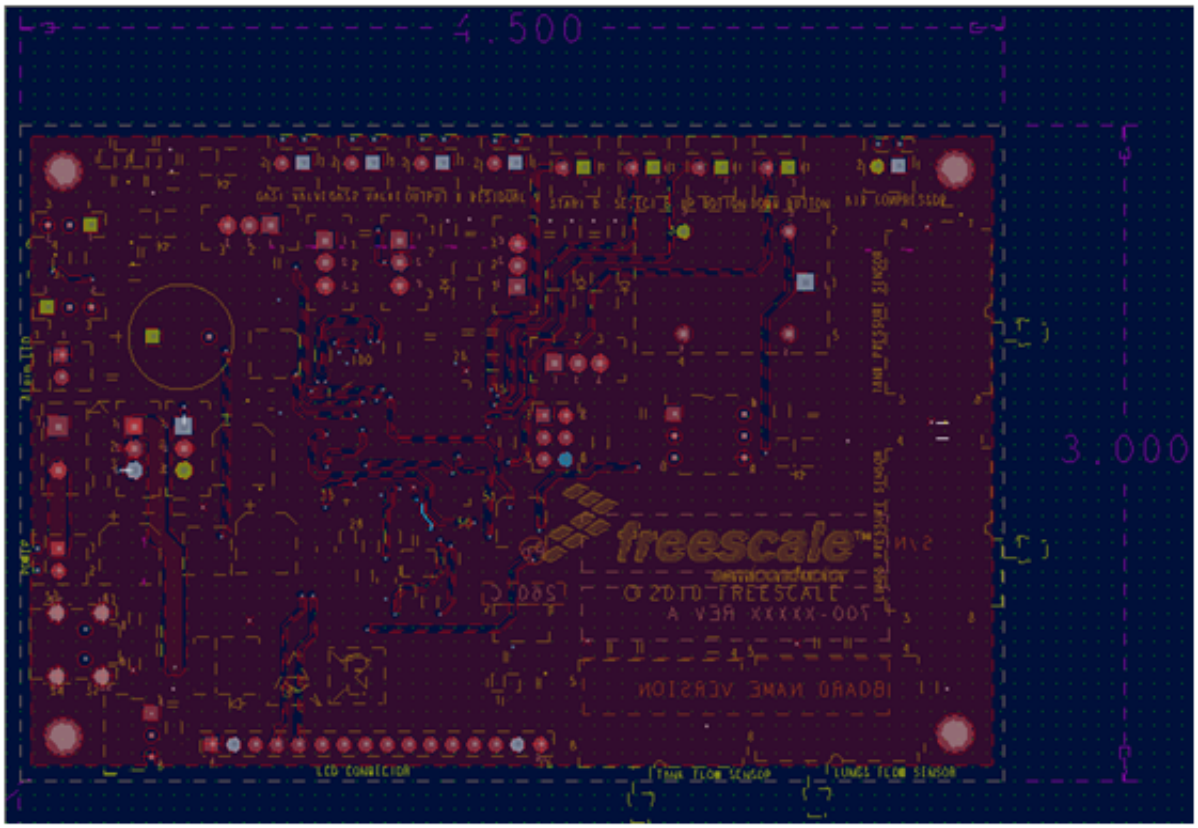

Figure 4-18. Ventilator layout design (bottom layer—routing and ground plane)

4.4.2 Physical PCB

After the PCB is manufactured according to gerber files, they look like Figure 4-19 and Figure 4-20.

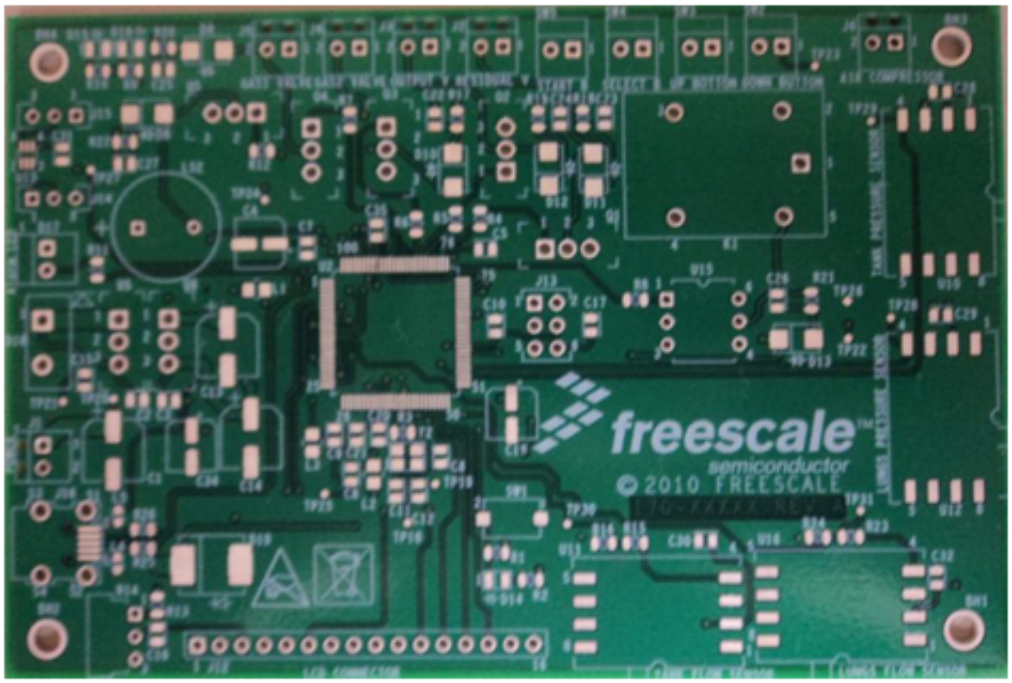

Figure 4-19. Ventilator PCB top view without components

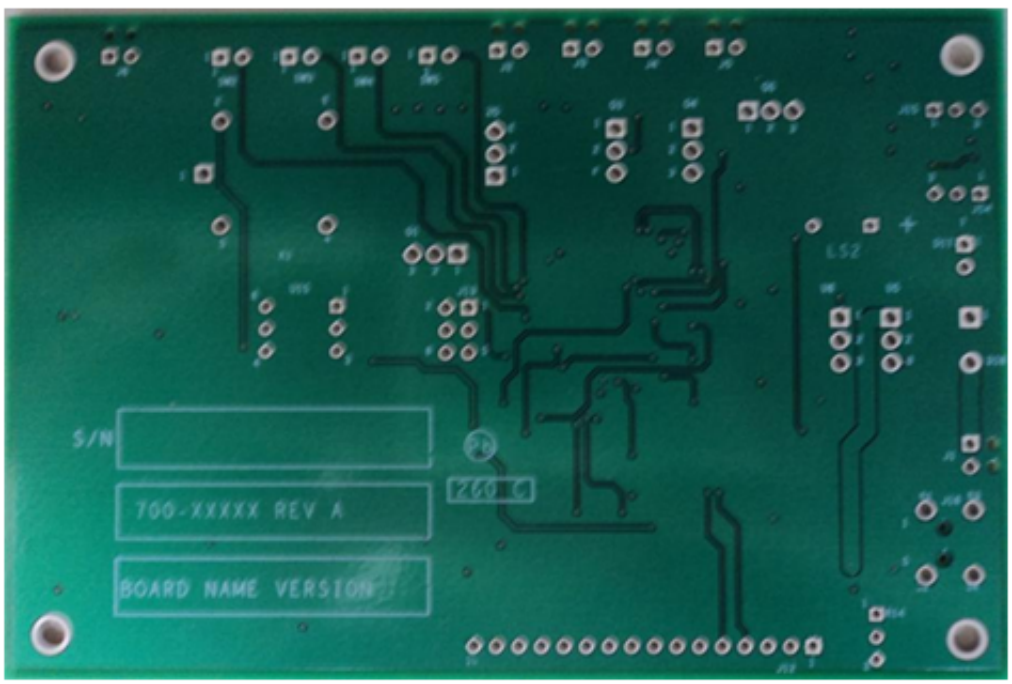

Figure 4-20. Ventilator PCB bottom view without components

After the components are assembled the PCB looks like Figure 4-21 :

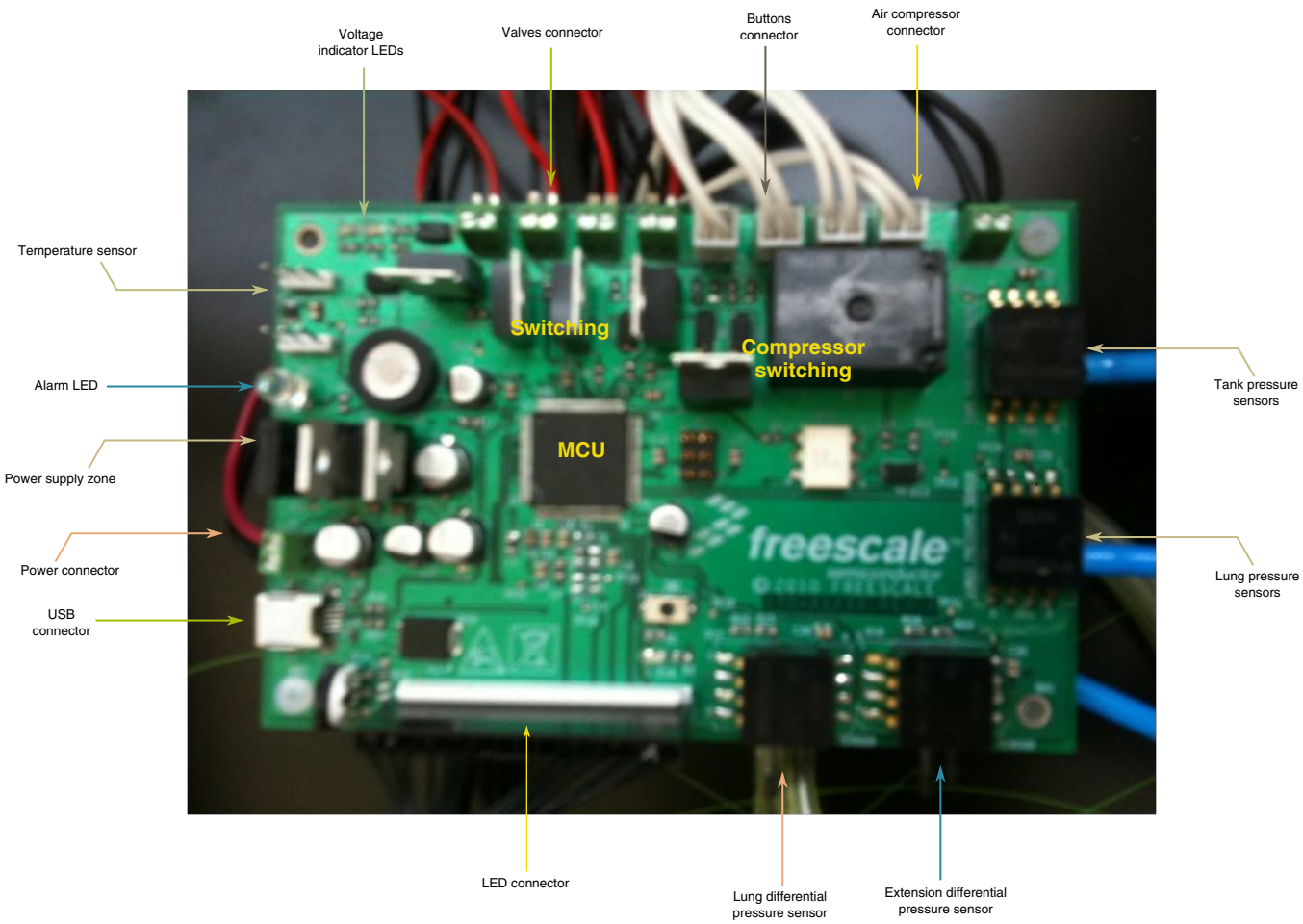

Figure 4-21. Real medical ventilator PCB

4.4.3 Ventilator Suitcase

After the PCB is properly working , both systems (pneumatic and electrical) are integrated. The demo suitcase shows the complete function of the system and Freescale component capabilities. The following images show the whole system integration. The following images show system integration.

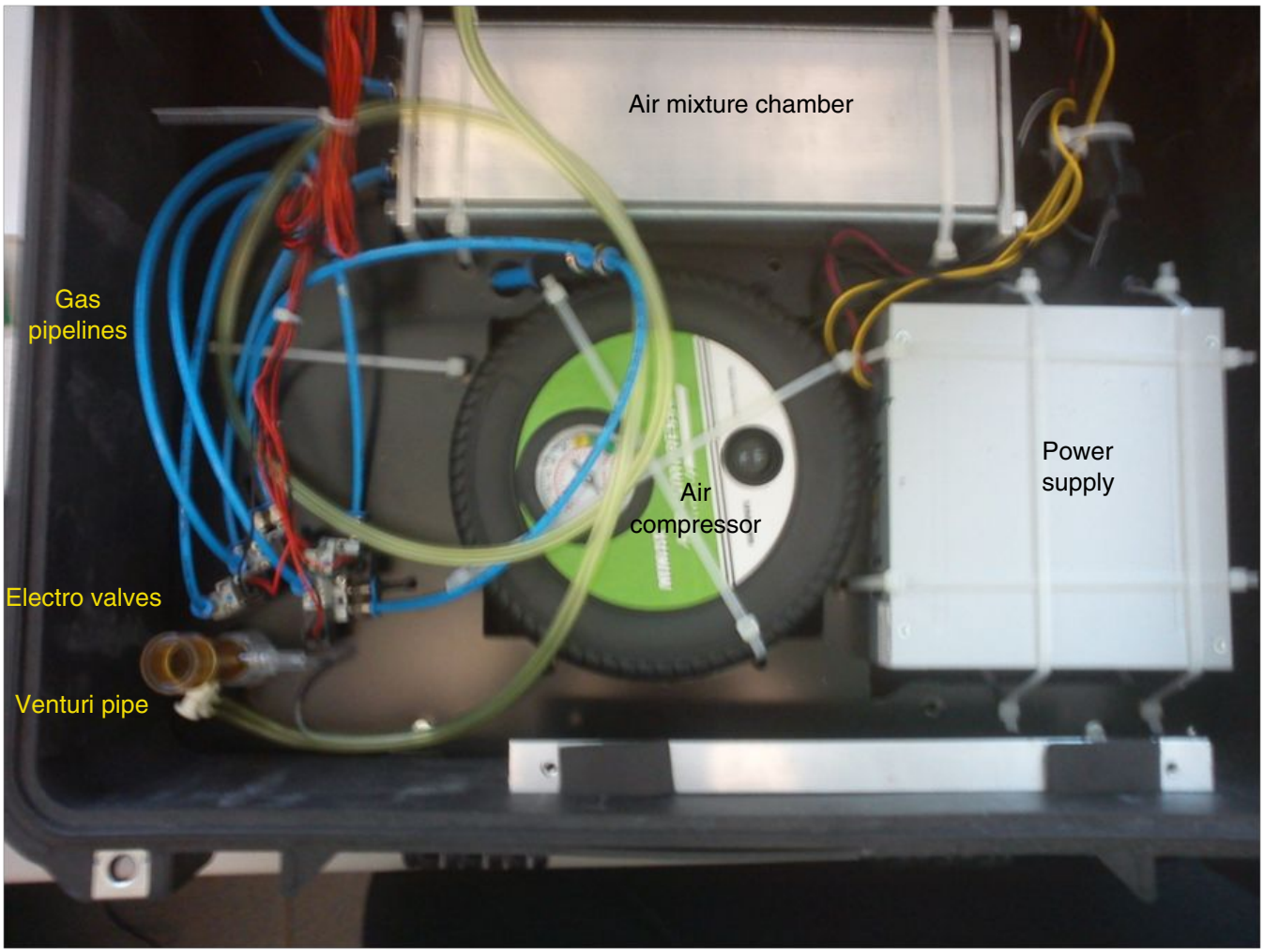

Figure 4-22. Ventilator bottom panel

Figure 4-22 shows the components for the medical ventilator. These components have been listed in the BOM. Figure 4-23 shows all the components under the principal panel.

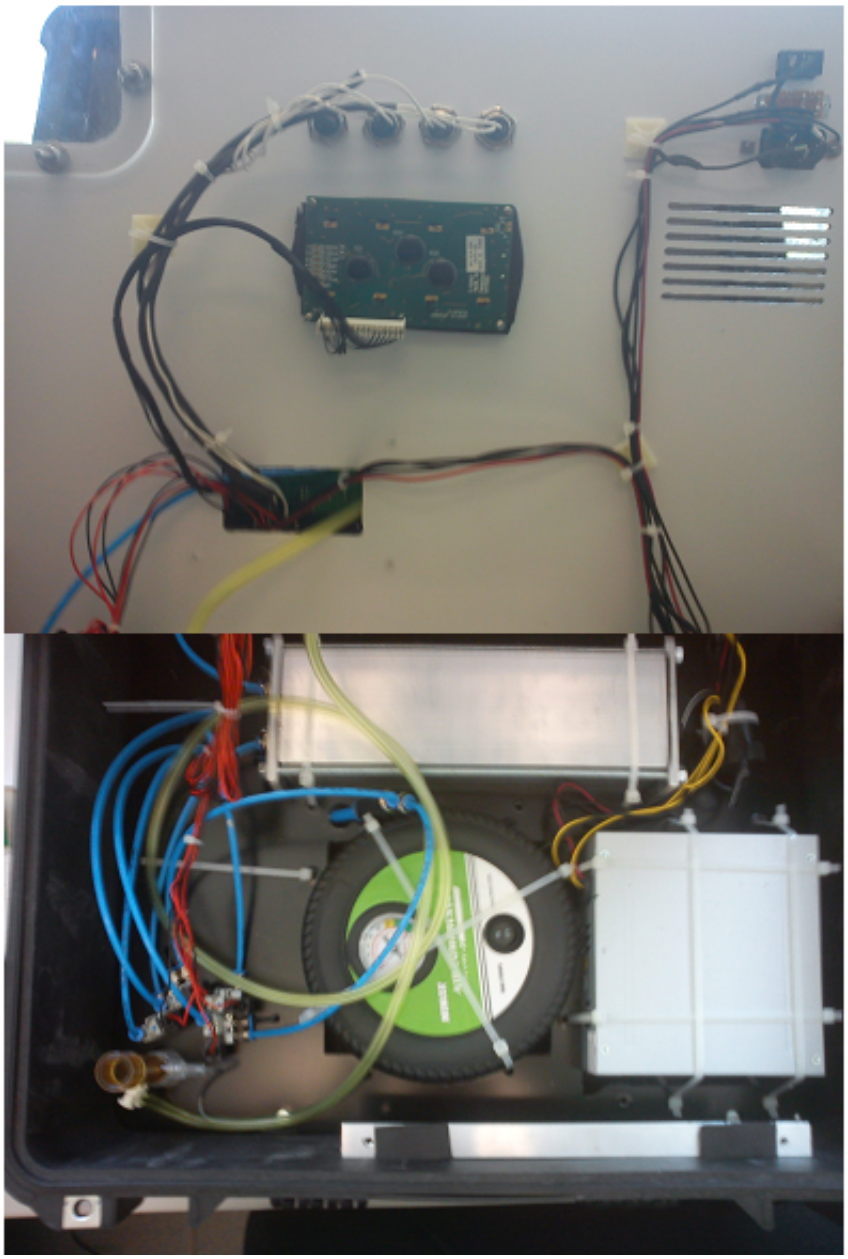

Figure 4-23. Ventilator inner components

Figure 4-22 shows the main panel final integration. The main objects are the electronics system in the PCB, the HMI, and the anesthetic bag with volume similar to children's lungs (proximally 1.5 L). The air capacity can be controlled by the PCB and configurable by the HMI.

Figure 4-24. Ventilator front panel

The next figure shows the complete suitcase, which measures 20.62" x 16.87" x 8.12" (52.4 x 42.8 x 20.6 cm)

Figure 4-25. Complete suitcase

Chapter 5
Software Design

5.1 Introduction

The software for this demo reference design is not intended to be implemented on a real medical ventilator, but to briefly show the alternatives to several real cases.

5.2 Software system description

Demonstration software must comply with the following points:

- Capability to show all functional modes—Pressure, frequency, and assisted.
- Debugging signals—LED in this case will turn on when the system is active in certain control modes.
- Probe main events—Lungs low and high pressure limits, tank low and high pressure limits, air compressor turns off while the system is active.
- Configuration capabilities to change and probe critical values.
- Cyclical menus
- The user must confirm modifications to prevent any unwanted changes.

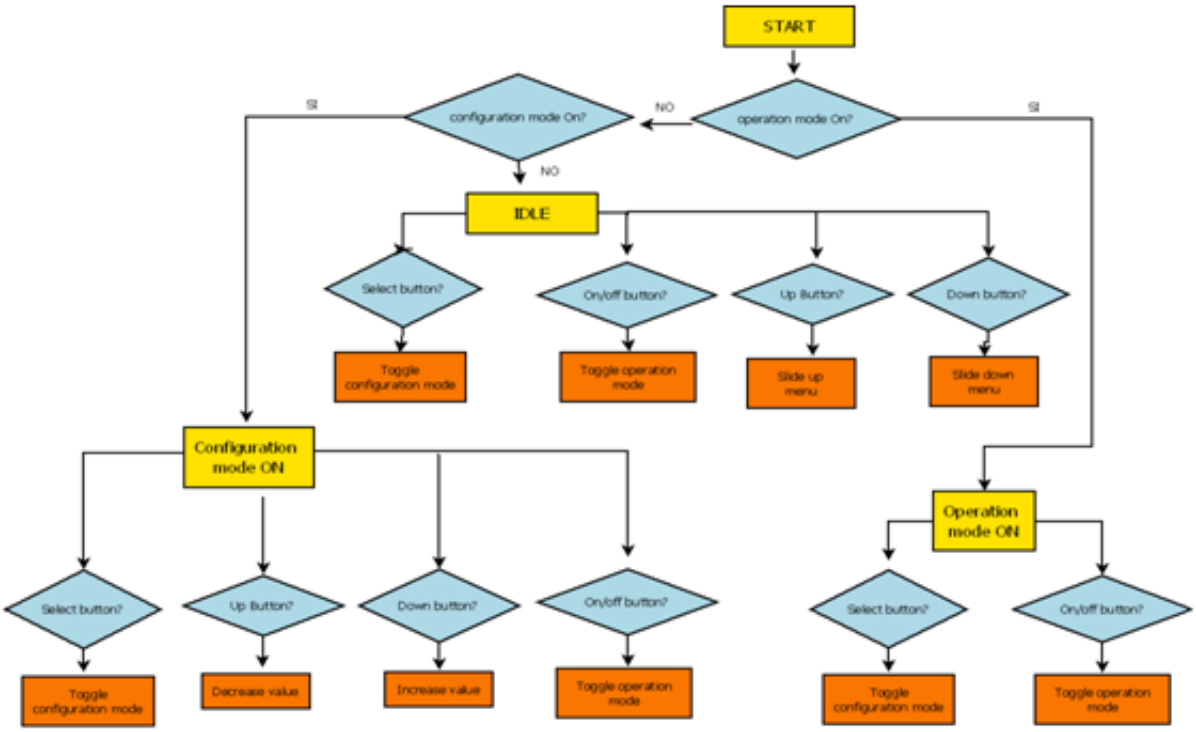

Figure 5-1. HMI ventilator flow chart

After the ventilator is in ON mode, the next flow charts proceeds:

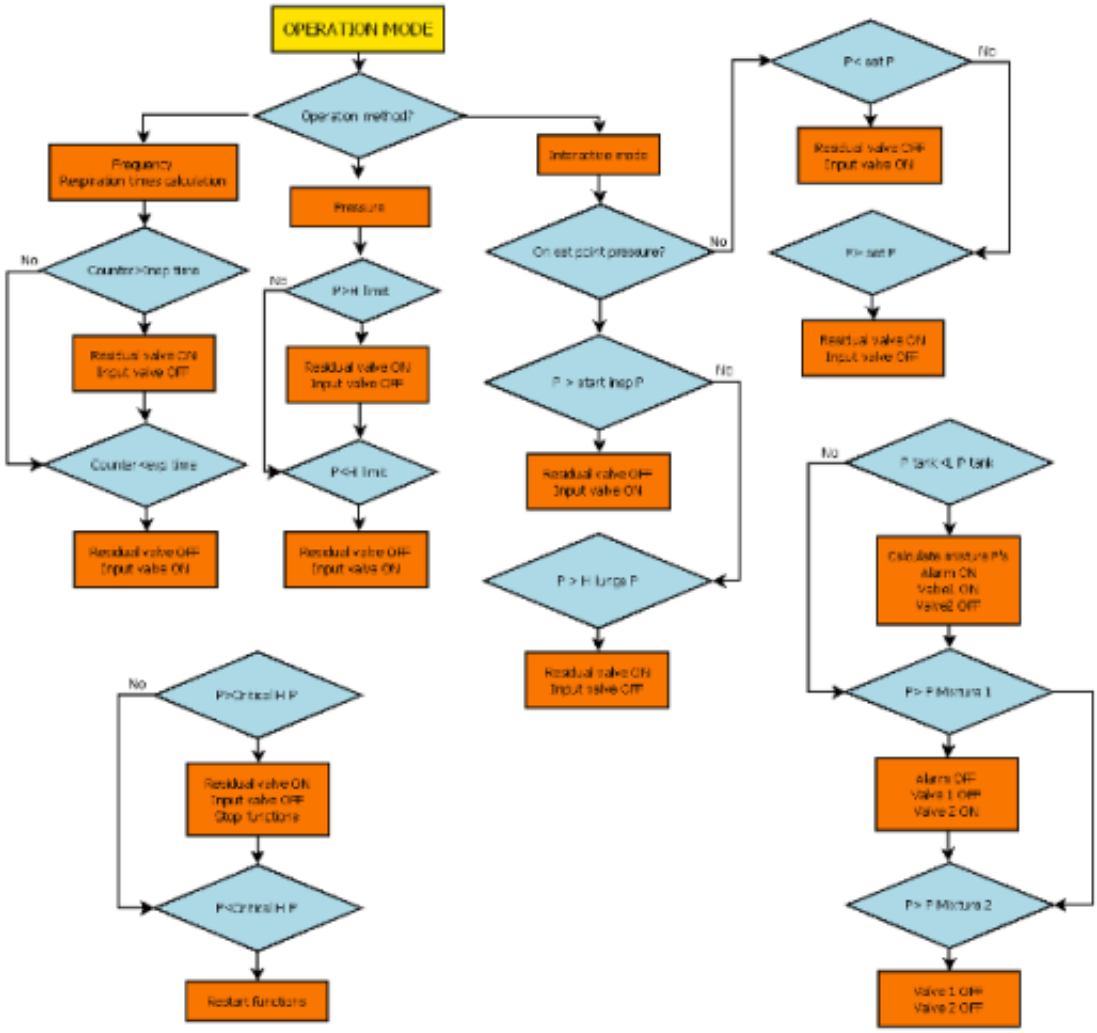

Figure 5-2. Ventilator function mode flow chart

5.3 Software architecture

After the software function description was developed, it is important to plan an organized and portable architecture.

In this case, Freescale C–SAR architecture was used, the following figure is a brief description.

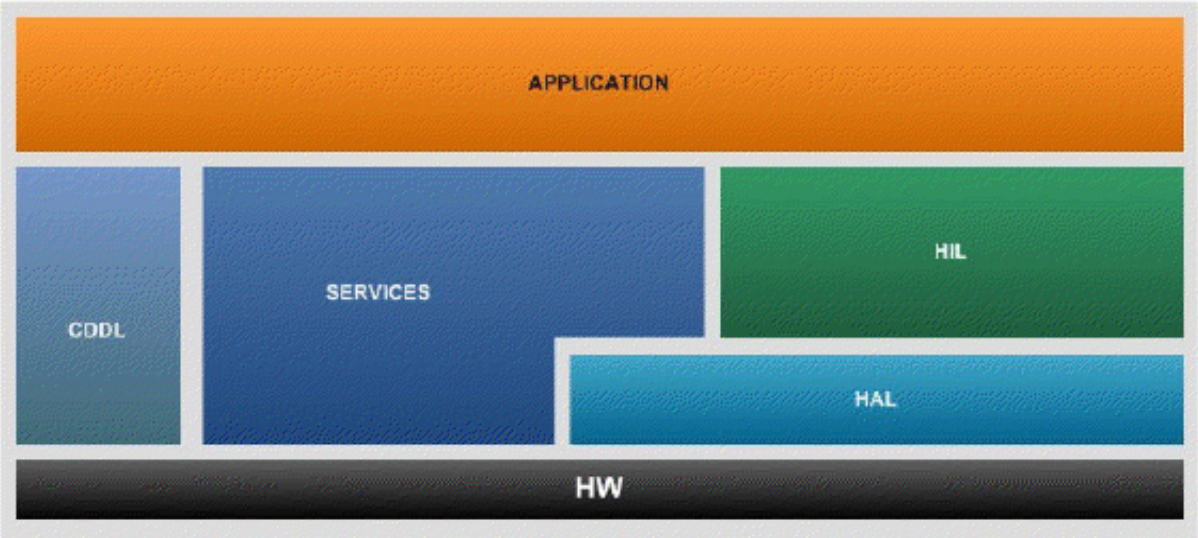

Figure 5-3. C-SAr reference model

For software model compliance, the CodeWarrior project was divided the following way:

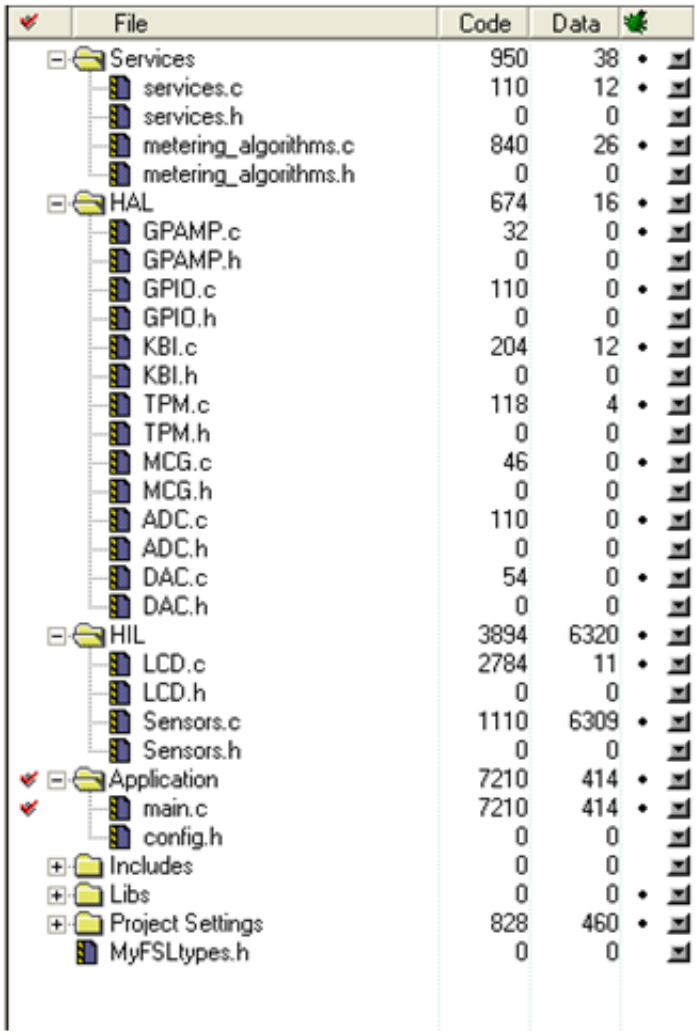

Figure 5-4. CodeWarrior Project structure

This software project structure is explained in the following sections.

5.3.1 Hardware Abstraction Layer (HAL)

This section is about the software made to drive the MCU modules. It reads values or registers, or write and configure registers perform certain actions or functions.

5.3.1.1 General Purpose Op Amp (GPAMP)

This is the MCU module that controls the internal Op Amps. In this case hardware configuration can be used as a buffer or as non inverter amplifier with a no external components variable gain.

5.3.1.1.1 GPAMP.c

void vfnInit_GPAMP(void)—This functions starts the Op Amp configuration, this configuration can change according to the gain chosen, Op Amp configuration, and input and output selection, for this case the DAC output as Op Amp input chosen use to obtain a non Zero reference for amplification.

5.3.1.1.2 GPAMP.h

Contains only relative header files, includes and function declarations.

5.3.1.2 General Purpose Inputs Outputs (GPIO)

This module configures I/O pins used to read or write on the pin chosen, like internal pull-ups for inputs or drive strength for outputs.

5.3.1.2.1 GPIO.c

void vfnInit_GPIO(void)—This function defines pin or port directions (input or output) and sets intial values. In this case valves, air compressor, LCD, buttons and LED can be driven by I/O.

5.3.1.2.2 GPIO.h

Contains relative header files that include the function declaration and define nick names for ports to understand which pin and register is being used.

5.3.1.3 Key Board Interruptions (KBI)

This module is used to launch interruptions when there is a change in a pin. The interruption can be by level or by a rising or increasing edge. The use of GPIO internal pull-ups help save components.

5.3.1.3.1 KBI.c

- void vfnInit_KBI(void)—Here the MCU configures its KBI module for flanges and levels, and cleans registers to use.

- __interrupt KBI_ISR_VECTOR void isrKBI2(void)—This function is obtained at each KBI interruption. In this case it only changes values of global variables that correspond to the state of HMI buttons. These states are used in the HMI menu to manage program flow.

5.3.1.3.2 KBI.h

Contains relative header file includes, function declaration, and defines nick names for ports, pines, interruption vectors, and configuration masks.

5.3.1.4 Timer/ Pulse-Width Modulator (TPM)

This module can be useful in two ways. For using Pulse Width Modulation (PWM) channels, where each one manages its own bandwidth and period, this module can also use the channel as a timer without a dedicated pin. In this case this timer is used by the whole system for synchronous tasks.

5.3.1.4.1 TPM.c

- void vfnInit_PWM(void)—In this function the PWM is initialized and configured for the clock divider, period, mode and interruptions. To use the PWM, the pulse width register must be modified (TPMXCnV).
- void vfnInit_Timer1(void)—In this function the TPM channel is configured as a timer with a certain frequency base, in this case for 0.250 ms.
- __interrupt TPM1_OVFL_ISR_VECTOR void isrTimer1(void)—This interruption is launched when the period base has passed, in this case this timer will modify a global synchronous counter and a time based flag for LCD control.

5.3.1.4.2 TPM.h

Contains relative header file includes, function declaration and defines nick names for registers, configuration masks, interrupt vectors, and constant values obtain a certain frequency.

5.3.1.5 Multipurpose clock generator (MCG)

This module sets the clock reference for the MCU bus clock; it could be external or internal under some modes. Frequency for the internal reference clock is selected in this module.

5.3.1.5.1 MCG.c

void vfnInit_MCG(void)—This function selects the reference clock, if selected an internal clock sets its frequency.

5.3.1.5.2 MCG.h

It contains relative header file includes and function declaration.

5.3.1.6 Analog to Digital Converter (ADC)

This module configures the ADC which converts an analog measure to digital. It can be configured, and in this case it also needs to dedicate a function to read the channel.

5.3.1.6.1 ADC.c

- void vfnADC_Init(UINT16 gu16Channel_mask)—This function configures and prepares the ADC channel for use, it needs an input parameter to choose which ADC channel will be configured.
- UINT32 gu32ADC_Read(UINT8 gu8Channel)—This function selects a channel to read and make the conversion to digital values, in this case it is a blocker task

5.3.1.6.2 ADC.h

It contains relative header files includes, function declarations, configuration mask constants, and definitions for ADC channel numbers.

5.3.1.7 Digital to Analog Converter (DAC)

This module configures the DAC to a certain analog value from digital values. The MM microcontroller is able to communicate certain outputs with certain inputs, in this case DAC, which will be used as reference for internal OpAmps.

5.3.1.7.1 DAC.c

void DAC12_Vin_SWtrig(void)—Initializes DAC module and configures it. In this case also set fixed values to use as a voltage reference.

5.3.1.7.2 DAC.h

It contains relative header files, includes and function declarations.

5.3.2 Hardware Independent Layer (HIL)

5.3.2.1 Sensors

These files are used to manipulate the values measured by the sensors, using the ADC and digital filters, performing the respective conversions.

5.3.2.1.1 Sensors.c

INT32 gs32FIR_Low_Pass_Filter (INT32 lu32Data, INT32* X_lp_out, INT32* X_lp)—This function is used to improve the signal measured from the sensors taking care of last measurement, and making the rate of change slower in case of a bad measurement. shows filter function blocks.

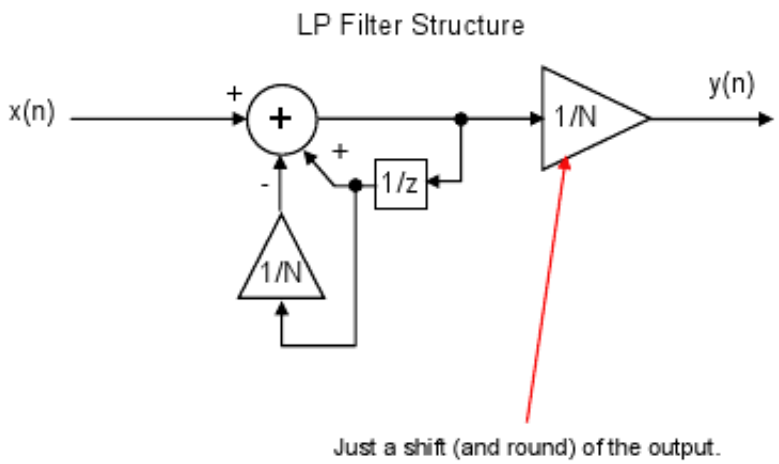

Figure 5-5. Low pass Function block and function description

With this filter the band width or frequency cut off can be defined with the following equation:

Eqn. 10

$$BW(\text{Hz}) - (SR(\text{Hz}) * (1/N))/(2*\pi)$$

Where SR is the sample rate of the MCU, N is the number of samples.

It can be used in the following way:

X_lp[1] = lu32Data + X_lp[0] - X_lp[0]/K_LP;

X_lp_out[1] = lu32Data + X_lp - X_lp/K_LP; //X_lp_out[1]=lu32Data/K_LP + X_lp_out[0]- X_lp_out[0]/K_LP;

- UINT32 gu32Temperature_Measure(UINT8 lu8_sensor_mask, UINT8 lu8_Sensor_Model)—This function calls the ADC port to read and convert the temperature value to mV. According to the sensor model and configuration of jumpers 1 and 2 (gain selectors), it sets the correct formula to obtain the temperature in °C.
- UINT32 gu8Flow_Volume_Measure(UINT32* lu32flow, UINT32* lu32volume, UINT8 lu8_sensor_mask, UINT32 Desired_Volume) —This function first gets the ADC value from its respective port and converts it to mV. Flow direction can be detected considering that in a DP sensor middle voltage corresponds to a pressure of 0PA. After direction is known, the next absolute values are considered in that direction, until it changes. Instant values are for flow and incremental ones are for volume.

The following equation is then applied:

Eqn. 11

$$Q = A_2\sqrt{\frac{2}{\rho\left(1-\left(\frac{A_2}{A_1}\right)2\right)}} * \sqrt{\Delta P} = K_{sensor} * P_{gas} * \sqrt{\Delta P} = K_{system} * \sqrt{\Delta P}$$

Where K is the sensor factor value, you have the total pressure for x measurements. It then needs to be multiplied by the CAPTURE_TIME factor to convert it to volume (ml). This value indicates the volume goes in same direction, and finally the function returns the direction and the application layer uses it.

5.3.2.1.2 Sensors.h

It contains relative header files, includes and function declarations, along with the next definitions:

- SENSOR_FACTOR—It is the K of the system, it means the relation between pressure and flow for this specific system.

CAPTURE_TIME—It is the time that takes to do a measurement. Definitions for pressure and temperature sensor models. Definition of measurement states (pressure and volume). Definition of sensors limits.

5.3.2.2 LCD

These files contain the main LCD driver for the general application.

5.3.2.2.1 LCD.c

This file contains the main controller driver for the LCD screen; here is a brief explanation of the general functions.

- UINT8 vfnClear_Screen(void)—Clean the screen with a simple call. It returns 1, if the screen is clean.
- UINT8 vfnInit_LCD(void)—Initialization for LCD screen, to configure the communication and data transmission. It returns 1 if the initialization has been successful.
- UINT8 vfnPrint_Line(UINT8 *gu8line, UINT8 gu8Size)—This function prints one line for a given number of characters and a pointer to the first data to be printed. It returns 1 when the string is printed and complete.
- UINT8 vfnBlink(UINT8 Cursor_State)—Function that blinks the characters in the cursor position that is referenced with its input variable Cursor_State. Returns 1 if blinking is working.
- UINT8 vfnSet_Line(UINT8 gu8address)—Set LCD to a horizontal line to print on the selected line.

5.3.2.2.2 LCD.h

Contains relative header files, includes, states definition, and function declarations for LCD.c.

5.3.3 Services

5.3.3.1 General Services

Files containing the general functions for application services, like discrete controllers, bit operation functions, and delay functions.

5.3.3.2 Services.c

This file contains the following functions.

- UINT32 ug32Negative_To_Positive(UINT32 u32Negative)—This function returns the complement for a negative number to get the actual value of the variable.
- INT32 s32_PID_Function (unsigned int Actual_P, unsigned int Desired_P))—This function acts as a discrete PID which can be called periodically to control one plant. For inputs you need a reference, the actual read value, and for outputs get the compensator for the controller.

5.3.3.3 Services.h

Contains relative header files, macros, includes, and function declarations for services.c.

5.3.4 Measurement and conversion services

These files help the application make some calculations about unit conversions.

5.3.4.1 Metering _Algorithm .c

This file contains some algorithms used for metering or for special operations.

- word SquareRoot(dword A)—This function returns square root calculations.

5.3.4.2 Metering _Algorithm .h

Contains relative header files, includes, and function declarations for Metering_Algorithm.c.

5.3.5 Application

This layer is where all the system functions are applied to give a particular function.

5.3.5.1 Main.c

Initialization section:

All files from HIL and service layers are included to call the files. Global variables are also defined, menu variables, and the constant definition of menu messages. There is a section for all variables where the value can be configurable through the GUI.

Before entering the infinite loop, this program calls all init functions in the lower layers like MCG, PWM, ADC, KBI, GPIO, GPAMP, interruptions, and the LCD for the system to use. After init the LCD system performs a welcome message sequence using the proper order of LCD functions.

Measurement section:

Inside the infinite loop system it first measures the temperature to make correct pressure measurements, then it refreshes both pressure values (lungs and tank) to show and have adequate control. After each measurement it prepares an average to show the user.

The differential pressure sensor can output flow volume and flow direction to a certain flow over a minimum point. After this measurement it can calculate the average for instant flow and by using the flow direction can add or subtract the volume quantity to show the volume delta after the system starts. This happens because this system does not include a method to measure initial volume, therefore the measuring flow uses the difference, which can be enough to take care of most patients.

Menu section:

After a certain time, values on the display are refreshed with considerable speed. The data shown is transferred by pointers to always show two values and the Freescale demo message. This menu has to be round cyclic, depending on the value of the variable lu8Active_Menu.

Inside this menu each parameter shows the parameter name, value and unit, or just the name and value, so each parameter divides the line in the appropriate number of characters for each part of the message. There are some parameters able to change its units, and with a predefined quantity of possible values. There are message selectors to these cases.

In this section, the menu performance is dependent on the system mode. In configurable mode there is a cursor on the first character that blinks, this indicates that it can be changed.

Functional section:

The functional mode for the ventilator can be separated in two parts, the tank mixture control and patient respiration control. This system has a unique way to control mixture, it has three methods to control breathing.

When the start state variable is TRUE, the system is on, and the MCU tries to have the mixture within an acceptable pressure range. If it is out of this range it has to allow more mixture in or let out mixture, this depends on its needs. In case it needs more mixture a visual and auditory alarm will be activated—in an ideal case this never will occur—therefore it intentionally lets the pressure drop until the alarm gets activated and restarts the compressor, in an ideal case this is always on.

The MCU needs to control valves to fill it with the exact amount of gas using the mixture percentage formulas for pressure. The system monitors the desirable pressure level for each gas. The Tank_State is then equal to the MIXTURE_READY, in this case the patient respiration can be executed.

The three alternatives to control breathing in this current design are by:

- frequency
- pressure
- interactively

Frequency—Defines how many respirations a minute are needed and how long inspiration and expiration (in terms of percentage) take. With these values, it calculates how many milliseconds to wait for each breathing phase, when the timer counter reaches this value it makes the transition to the next phase

Pressure—Defines low and high pressure limits for respiration. The system continuously checks lung pressure. When the pressure comes down to a low limit, inspiration starts; and when pressure reaches the high limit, expiration starts.

Interactive—In this mode, the system gets the balloon pressure to the desired low limit and keeps it at the same value as possible. It then detects pressure increases to a certain value and completes the inspiration cycle until the correct high value is met.

These modes have two critical limits, a low and a high limit. First you need know to if a lack or leakage of gas exists, in this case an alarm is activated. The high limit is to avoid a pressure level that can be dangerous for the patient or in this case the balloon. If this happens it lets out exceeding gas and then comes back to normal operation.

You can modify any variable used as reference to control the system functions via the HMI.

Configuration section:

This section uses the Display_State variable that can have two values configurable or idle, if it is idle the buttons have the following results:

- UP—Increases the menu counter if it is on the highest return to the lowest, therefore modifying this counter in the menu section shifts the menu up.
- DOWN—Decreases the menu counter if it is on the lowest return to the highest, therefore modifying this counter in the menu section shifts the menu down.
- SELECT—Changes Display_State variable to configurable and turns on a flag to indicate to the menu, to turn on the blinking function.

After Display_State variable is on, the configurable state buttons have the following effects:

- UP—Effects depend on which parameter is modified, in some cases it can increase values. It keeps the highest value after it is reached. In other cases it goes cyclic or changes to the next state.

- DOWN—Effects depend on which parameter is modified, it can decrease values in some cases. It keeps the lowest value after it is reached. In other cases it goes cyclic or returns to the last state.
- SELECT—It turns off the blinking menu flag and changes the Display_State variable to an idle state. The system then refreshes values used when system is on.

5.3.5.2 Config.h

The Config.h file has all defines used in the main section and describes the limits, sensors, and all the variable limits that can be modified. There are some sections, explained below; the file can be divided easily.

Configuration states section:

General defines to configure or not configure the main parameters since the main menu. CONFIG_IDLE and CONFIG_PARAMETERS are the main defines of this short but important section.

Configuration Parameters Menu section:

This section defines the main parameters used in the main section to select parameter configurations.

This parameters menu is composed of:

LUNGS_PRESSURE
TANK_PRESSURE
OPERATION_MODE
TEMPERATURE
OXYGEN_PERCENTAGE
TANK_LOW_PRESSURE
TANK_HIGH_PRESSURE
ALARM_TIME
LUNGS_LOW_PRESSURE
LUNGS_HIGH_PRESSURE
INSPIRATION_PERCENTAGE
RESPIRATION_TIMES
PRESSURE_UNITS
TEMPERATURE_UNITS
MENU_LENGTH

Unit definition section:

This section defines the kind of units used for the pressure, and temperature units. Pressure units are defined as MPSI, PA, CMH2O, MMHG, and MBAR; for temperature units CELCIUS and FARENHEIT.

General configuration defines section:

General configuration defines are explained in detail.

- INSPIRATION_TIME—The percentage of time a person on the ventilator breaths, the initial value is 25.
- EXPIRATION_TIME—This value is the complement of the previous value, and must always sum 100%.
- RESPIRATIONS_PER_MINUTE—The frequency patient is going to breath, the initial value is 14.
- TIME_MODE_IDLE—Time mode idle defines for state machine.
- INSPIRATION_STATE—Inspiration state define for state machine.
- EXPIRATION_STATE—Expiration state define for state machine.
- LUNGS_LOW_LIMIT_PRESSURE—Defines the lower pressure that can be in an expiration, the initial value is 700 PSI.
- LUNGS_HIGH_LIMIT_PRESSURE—Defines the higher pressure that can be in an inspiration, the initial value is 900 PSI.
- LUNGS_HIGH_CRITICAL_PRESSURE—Defines the critical pressure that can be in any lung, the initial value is 1600 PSI.
- TANK_HIGH_LIMIT_PRESSURE—Defines the higher pressure that can stand in the pressure chamber, this initial value is 14500 PSI.
- TANK_LOW_LIMIT_PRESSURE—Defines the lower pressure that must be maintained in the pressure chamber, this initial value is 12000 PSI.
- SENSOR_HIGH_LIMIT_PRESSURE—Defines maximum pressure the sensor can measure, the limit for the sensor is 14000 PSI.

Operational Modes section:

This section defines the main operation control modes for ventilation (explained previously) PRESSURE_MODE, TIME_MODE, INTERACTIVE_MODE.

Sensors Used section:

In this section the config.h defines the types of sensors to be used for pressure measurement. It defines the LUNGS_PRESSURE_SENSOR as MPXV5050GP, the TANK_PRESSURE_SENS0R as MPXV5100GP and the TEMPERATURE_SENSOR as LM94021.

Conversion Factors section:

This section defines all the factors for conversions between all the possible units.

Chapter 6
Parameters Configuration

6.1 Parameters configuration

Main parameters configure and interact with users while the Freescale ventilator is working or idle. These parameters give the application software the control that is implemented, maximum and minimum tank/lung pressure, providing the ventilator with flexibility for a wide range of use. users. Main configuration variables and parameters are explained in detail.

Table 6-1. Variables and parameters

Name	Description	Program Variables and related defines	Possible Values (initial value)
Oxygen percentage	This is the quantity of oxygen the control lets flow inside the tank when the air and oxygen mixture is made.	Oxygen_percentage	0–100 (20)
Breathing frequency	It is the frequency at which the user breaths, in breathing times per minute.	Respiratory_Frequency defines: • RESPIRATIONS_PER_MINUTE	6–17 (14)
Tank Low/High Limit Pressure	Defines the range of pressure in the tank. If the pressure is lower than the limit, the compressor turns on to refill the tank to the pressure required.	lu32Tank_Low_Limit_Pressure lu32Tank_High_Limit_Pressure Defines: • TANK_HIGH_LIMIT_PRESSURE • TANK_LOW_LIMIT_PRESSURE	0–14500 (12000 – 4500)

Table continues on the next page...

Table 6-1. Variables and parameters (continued)

Name	Description	Program Variables and related defines	Possible Values (initial value)
Lungs Low/High Limit Pressure	Defines the range of pressure in the lungs. If the pressure is lower than the limit, the inspiration valve turns on after each period passes. Helps to control the air flow and pressure.	lu32Lungs_Low_Limit_Pressure lu32Lungs_High_Limit_Pressure Defines: • LUNGS_LOW_LIMIT_ PRESSURE • LUNGS_HIGH_LIMIT_ PRESSURE • LUNGS_HIGH_LIMIT_ PRESSURE • LUNGS_HIGH_CRITICAL_ PRESSURE	0–1600 (700 – 900)
Operational Mode	Selects the operational mode the ventilator will use. There are three modes of operation: pressure, time, and interactive mode.	gu8Operational_Mode Defines: • PRESSURE_MODE • TIME_MODE • INTERACTIVE_MODE	Any mode on the defines (PRESSURE _MODE)
Alarm Time	Defines how much time the alarm is going to ring when required.	Alarm_Time Defines: • BUZZER_TIME	0–65535 ms (1000ms)
Inspiration/Expiration percentage time	Sets the percentage of time that an inspiration and expiration takes of the breathing period defined previously.	Inspiration_Time Expiration_Time Defines: • INSPIRATION_TIME • EXPIRATION_TIME	Any complement percentage (25% – 75%)
Pressure Units	Defines the pressure units that are going to be used to interact in the user interface on the LCD.	lu8Preassure_Unit Defines: • MPSI • PA • CMH2O • MMHG • MBAR	All shown in the defines (MPSI)
Temperature Units	Defines the temperature units that are going to be used to interact in the user interface on the LCD.	lu8Temperature_Unit Defines: • CELCIUS • FARENHEIT	All showed in the defines (CELCIUS)

Chapter 7
Troubleshooting

7.1 Troubleshooting

This section tries to attack main issues that may happen if the system fails. This section tries to reduce the impact of some common issues the ventilator may have, like system errors and fails.

7.1.1 Main issues

The ventilator does not turn on when the switch is on—This issue occurs frequently when the user does not have an idea on how to get start with the Freescale demonstration ventilator. This issue is easy to fix and takes less then five minutes. To solve this issue try the next steps:

1. Check that the power cord is connected to the main power source in the suitcase, and connected to AC voltage.
2. Check that the selected voltage is the correct voltage supplied to the ventilator power source. The standard value for this is 120 Vrms, but verify if your electricity supplier has this standard.
3. Listen to the power source ventilator to check if the main board has an issue or the power source does not work (you may want to unscrew the ventilator to check this).
4. If the previous steps did not work, unscrew the main tap and verify that the power supply is connected to the main control board. See Figure 7-1 to get better idea on how the cable must be connected.

Figure 7-1. Power supply connections

Emulator does not completely fill or does not fill at all—Sometimes, even with the sound of air flowing away from the compressor chamber, the lung balloon does not fill completely.

This issue occurs mainly when the ventilator has been used many times, or has been transported in not such good condition. To solve this issue try the steps:

1. Unscrew the main panel to reach the main tubes and valves that control air flow inside the ventilator suitcase.
2. Check that the tubes inside are properly connected, if you see disconnected tubes you can connect them as shown in Figure 7-2.

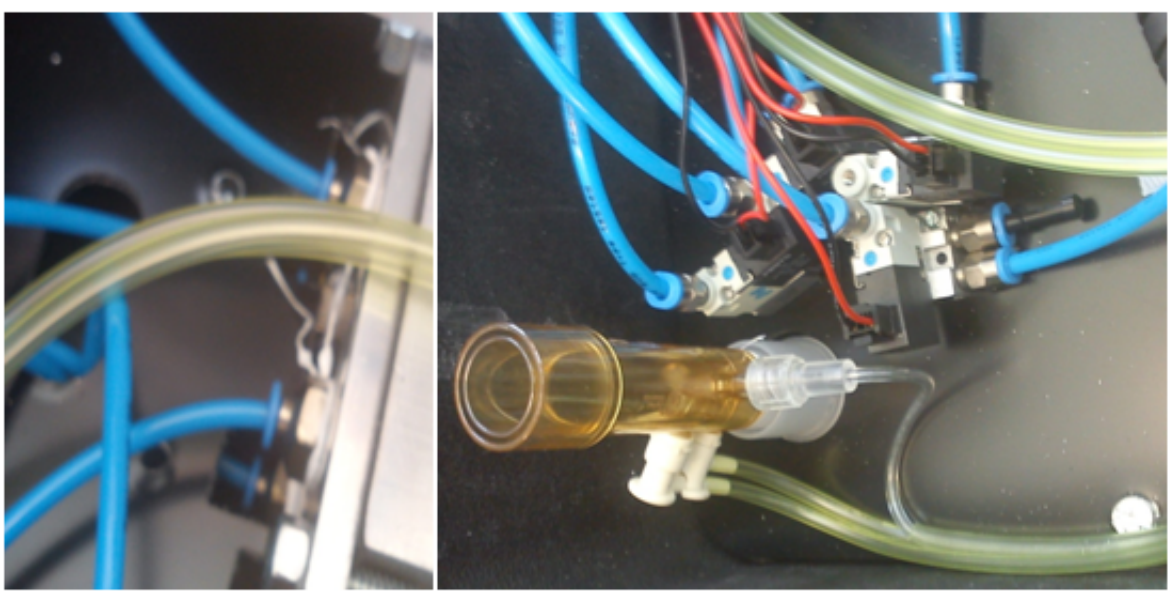

Figure 7-2. Valves and hoses connections

LCD issues —Display characters sometimes can be wrong or the LCD screen does not turn on properly. To solve this issue try the next steps:

1. Check that the LCD connector cables are plugged in correctly as shown in Figure 7-3.

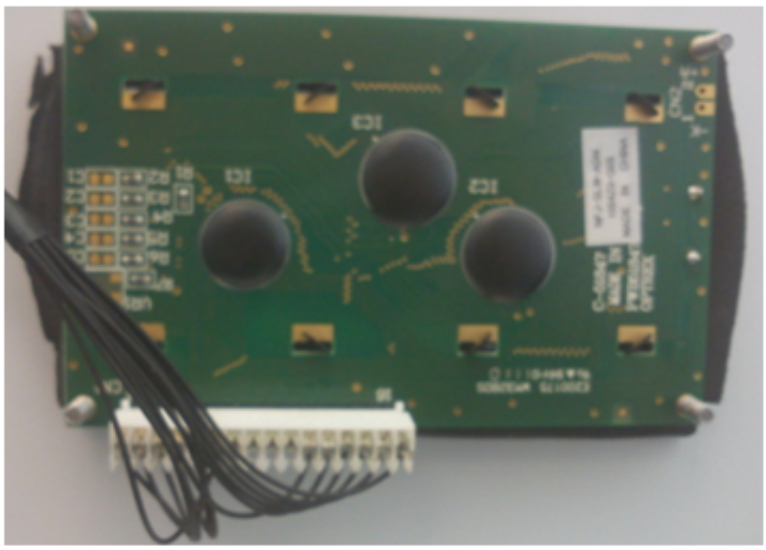

Figure 7-3. LCD connections

2. If the LCD is showing some errors in its display, you must place the suitcase far way from electromagnetic fields such as microwaves, induction stoves, and some cell phones. Then reset the entire demo with the on switch without handling the suitcase.

Compressor does not turn on—Some times the compressor does not turn on the first time you push the start button, this issue can be solved with following steps:

1. Check if the power source is on and that the main board LED is also turned on. If everything is on, check the power connections that go from the main PCB board to the compressor. These connexions are shown in Figure 7-4. The black cables are the compressor control switch from the main PCB.

Figure 7-4. Power connections

2. If the previous example does not work, unscrew the main panel and check that the compressor power switch is turned on. If not turn the switch on, if this still does not work, change the entire compressor to verify if the main board works.

Plastic lung emulator has erratic behaviour—This issue is due to improper connections in tubes and valves. To solve this issue try the next steps:

1. Check that the pressure sensors in the main board are properly connected, shown in the next section “Main board connections”.
2. If the previous step did not work, unscrew the main panel and check for the electrical valve connections shown in the figure. All valves need to have signal (red wire) and ground (black wire).

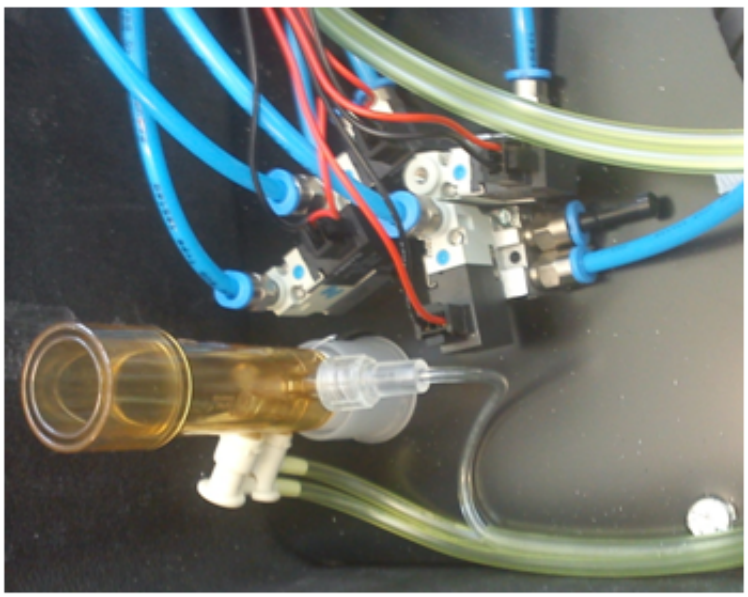

Figure 7-5. Valve connections

7.1.2 Main board connections

This section demonstrates how to review all the important main board connexions and check if they are connected properly for the ventilator to function properly. Figure 7-6 indicates areas that connect the main control board to the user interfaced peripherals, and the devices that interact directly with the application such as, valves, compressor, and flow measurement devices.

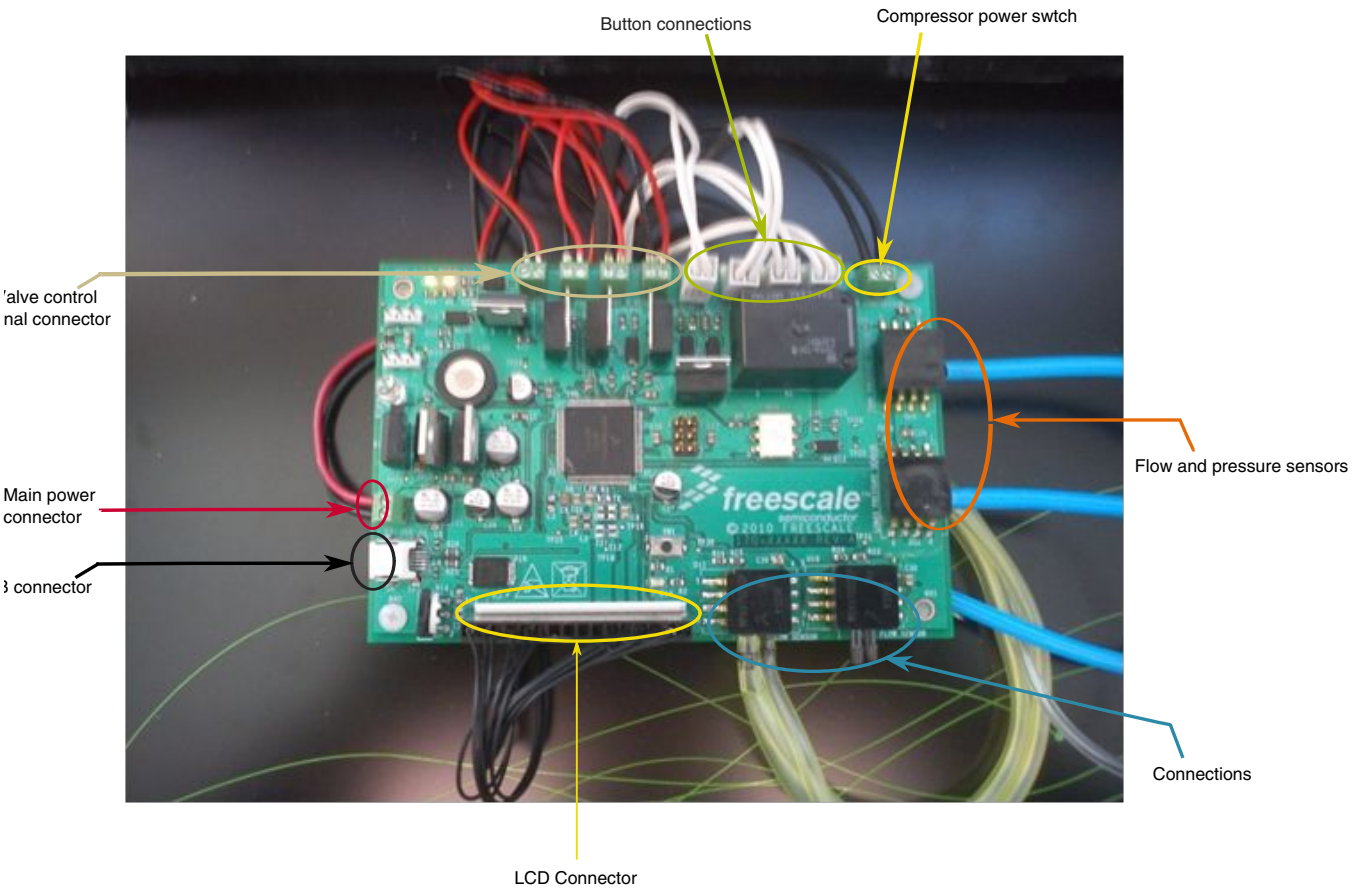

Figure 7-6. Main board connections

Button connections—These connections go inside the suitcase. They need to be connected to control the application through the user interface. If some cables are unplugged, the buttons will not respond.

Compressor power switch—This connector switches the power of the compressor to reach in the air chamber the desired pressure. If this connector is disconnected, the compressor would not be on.

Flow and pressure sensors—These sensors are connected through tubes that carry pressure inside them. If any of these sensors are disconnected the application could not work properly. Note that not all sensor must be connected a minimum of three are needed and may be extended for future control needs.

LCD connector—This pin out connector sends the main instructions and power to the LCD board. If this connector is disconnected, the LCD interface will not work.

USB connector—USB connector works as an application interface with other MCUs or directly with any computer interface.This connector does not affect the functionality of the ventilator, and the interface can be developed in the future.

Main power connector—Power connector supplies the power to the board to accomplish all the main tasks. This connector must be plugged in at all times.

Valve control signal connector—This connector must always be plugged in to control the main valves for the ventilator applications.

Chapter 8
Conclusion

8.1 Conclusions and References

This complex and critical medical system for patients demonstrates the effective use of Freescale components.

- Pressure sensors allow accurate and precise measurements that are required for this kind of medical application.
- The MCF51MM MCU is an ideal alternative for such applications; it has enough processing power, memory, and speed. Their interfaces (USB) are one of the best alternatives for the medical industry and its cost effective analog modules. These modules allow flexibility to improvise and experiment with new settings without changing the PCB layout.

References

- MCF51MM256 Reference Manual (document number MCF51MM256RM)
- TWR–MCF51MM Schematic
- TWR–SER Schematic
- MP3V5050
- MPXV7002

These documents can be found on the Freescale semiconductor web page, www.freescale.com

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: [No Doc ID]
Rev. 0, 11/2011