

Ever wonder how to put together a robot or what goes on before the competition starts. The TALON 540 Team has put together some information to get you started on “what’s behind the scenes”.

Game Strategy by Meher M.

TALON 540, or the Mills E. Godwin Robotics Team, is a team focused on exemplifying the ideals of FIRST (For Inspiration in Science and Technology), a company that is designed to give real-world experience at the high school level. Each year, we are assigned to build a robot that will play in a game designed by FIRST mentors and executives. The team then has six weeks to build a robot, after which the robot is shipped off for competition. TALON 540 is somewhat like a microcosm, divided into many different parts. One of the most important parts of our team is strategy.

The game introduced to us this year, called Lunacy, is a game played on a 54x27 ft field. In honor of the 40th anniversary of NASA’s landing on the moon, the entire field is covered with regolith, simulating the effects of driving on a surface with 1/6 gravity. The game pieces used for Lunacy are called moon rocks, empty cells, and super cells and are made out of toy orbit balls. These game pieces can be thrown by a human player, or shot off by the robot (dependent on strategy). During each match, three teams play on one alliance, while another three teams play for another alliance. The object of the game is to place as many moon rocks into the opposing alliance’s trailers, which are attached on to each team’s robot. Each match is 2 minutes and 15 seconds long, consisting of two different periods, autonomous and teleoperated. The autonomous period is where the robots are programmed to move by themselves, and if possible score points. The

teleoperated period is when the drivers can take full control of their robots. Points are scored by placing/shooting a ball into another opponents trailer. Moon rocks and empty cells are worth 2 points, while super cells, which can only be scored during the last 20 seconds of the game, are worth 15 points. Each alliance only gets a certain number of empty cells, which can be placed in a fueling station (area on the corner of the field) and then be exchanged and used. Three human players are used per each alliance, 2 located near the fueling stations, and one located on the middle side of the field. Human players are designed to shoot the game pieces into the opponent's trailer along with the robot. The game this year is very complex, and requires a lot of teamwork.

The strategy team this year, as shown above, plays an enormous role in the game this year. The team is the heart of the competition, training drivers, and constantly preparing for anything that is to come during competition. The team picks drivers and teaches them how to drive efficiently during competition and under pressure. The last aspect, competition strategy, is by far the most important. Each year, based on the game, the lead strategist, his/her panel of team members, and a mentor come up with a play book, or a set of strategies based on almost any possible situation that the team may encounter during competition. This playbook is usually made during build season along with the construction of the robot.

So far, the team has completed two sets of vital tryouts, one for human player, and the other for drive team. The drive team tryouts, which took place at the beginning of the year, are a great way of getting ready for build season. The team picked four pairs of drivers, which will be used and trained during build season. Next, the team has completed a human player tryout, where they threw moon rocks into moving trailers.

The team has already picked a couple of potential human players, which will be an important part of this year's game. Still waiting for the design to be completely finalized, the strategy team has not yet come up with a comprehensive playbook.

The strategical part of TALON 540 this year is more important than ever. Regardless of the outcome of the robot design and efficiency, if our strategy or method of playing the game is well thought out, we still might have a shot at winning regionals this year.

Design by Ryan A. and Brock D.

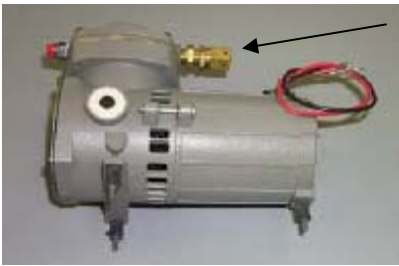
When one sets off on the journey that is building a robot, one must first consider the task that the robot is to perform. After this consideration, the next step is to come up with a way to carry out the assigned task as efficiently as possible. With the use of motors and pistons, along with simple machines, almost any task can be accomplished. The next step is prototyping, in which the designs are built in order to prove or disprove their effectiveness. Often, the prototypes are built out of spare materials and are not the best-looking as they are only proof of concept. If the design does not prove effective, one must go back to the drawing board to design something else. However, if the prototype does work, the measurements and parts needed to build it for real are taken down so the finished product is close to perfect and can be used on the robot. The design process, although lengthy, is a deeply rewarding experience in that one has solved a problem and executed its solution.

CAD by Greer P.

Autodesk Inventor, a 3D parametric modeling software, is a useful tool which can be used to facilitate the design and building of a manufactured item, or in this case, a robot! The first main thing that you need to do in order to CAD something is to decide what it is that you are planning on designing. Once you have decided upon what to design/build, you can get to work. By utilizing basic dimensions of what you are planning on building, you can implement them into the software, and design a virtual version of what you need, and even run it through stress tests to see how it will probably perform under multiple conditions. Changing the size and dimensions of the part will also update itself in any assemblies that it is included in, and you can change more than one part, real-time. Inventor is a wonderful instrument to utilize in the designing of our robot, and has facilitated the completion of our machine.

Pneumatics by Andrew S.

Mount the Compressor with the vibration isolation mounts. The compressor will put out approximately 120psi before the relief valve opens. You should attach the relief valve to the back part of compressor.



Attach the pressure switch to the system this will allow you to turn off the compressor once you are up to 115psi, saving battery power.

In the system, you can attach up to four tanks, more tanks more air but also adds more weight.

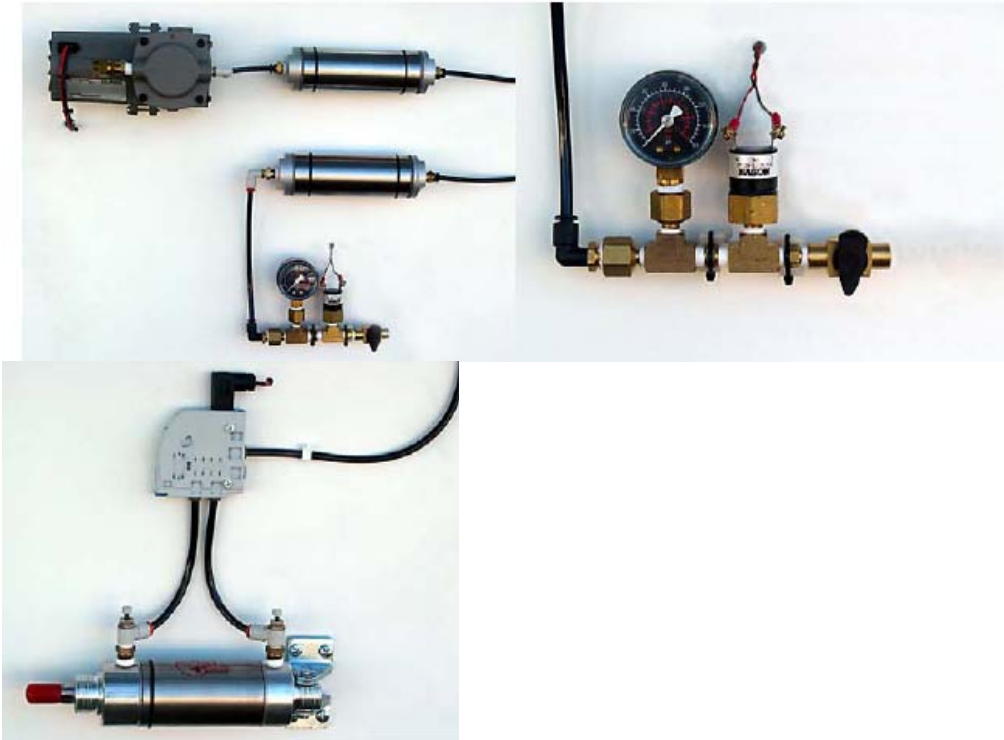
Your system also needs to have a plug valve to manually release the system and a Pressure Gauge to read the pressure in the system

If you need to have different air pressures leaving the system you can use a Regulator to regulate the pressure coming out of it the regulator should have a Pressure Gauge on it.

There is a Festo Solenoid provided which goes after the Regulator in the system and is used to control whether the Bimba is released or not. You can have more than one solenoid on the same regulator if you want them at the same air pressure.

The most important thing is that when you attach fittings together always use Teflon tape to prevent leaks.

Here is an example of a pneumatic system.



Electronics by Chris L. and Teja C.

Looking at the jumble of wires on a robot's electronics board, you may think electronics is difficult. It is, if you decided to get in-depth with the subject, but even beginners can make a passable e-board by following simple directions and using organization and logic.

As an electrician, you control the path electricity takes around the e-board. Before you begin, it is essential that you coordinate with the mechanical and design so they know what your space requirements are and if any of their constructs will come in conflict with your wiring.

After getting your final dimensions and alerting the design/build teams to your needs (space for a battery, placement of the main kill switch, circuit board, wire clearances, etc) sketch out your board's plan, to scale if possible. A key to a successful wiring is organization; keep wires as clear, straightforward, and organized as possible.

- Plan ahead – will there be clearance or access issues?
- Stay organized
- Keep it neat
- Keep it clean
- Label and color code everything
- Assume stupidity (even on your part!) and, from there, make it safe.

Training and information can be found on the FIRST website, the current competition manual, and on ChiefDelphi.com.

As you build, you may hear confusing terms thrown about. Below are essential concepts to help you understand the lingo:

- Current – the flow of electric charge, measured in amperes (“amps”). It measures the **quantity** of electricity per time. (Compare to mass flow rate in hydraulics)
 - Amps (ampere; amperage) - amount of electric charge per unit time, measured in coulombs per second.
 - $1 \text{ Coulomb} = 1 \text{ Amp} * 1 \text{ second}$
- Volts (from “voltage):
 - The **rate** at which energy is drawn from a source, producing a flow of electricity in a circuit
 - OR
 - The difference in electrical charge between two points in a circuit
 - Similar to a water pressure difference in a water tower.

- The higher the tower, the faster the water will flow out the bottom.
- In an analogy to water, voltage can be compared to the pressure/speed of the water in a pipe, and current can be compared to the mass of the water that moves past a point in one second. A water-cutter has high speed but low water mass (high voltage, low current), while a waterfall has low speed but high mass (low voltage, high current). Both speed and mass can kill you if they get high enough, and so can voltage and current.
- Ohms (Ω) = resistance = volts / amps (I)
 - $\Omega = V/I$
 - $V = \Omega * I$
 - Wire resistance
 - (The wire's resistance constant * length) / wire's cross-section area
 - $R = (P*L)/A$
- Electrical power is measured in watts; equals (volts*amps)
- Diode: a specifically-calibrated resistor that allows current flow in only one direction; like a light-bulb, without the light-up function.
 - Resistor: something that causes a voltage drop; a load
- Wire gauge: diameter of the wire. Similar to shotgun gauge, the bigger the wire, the smaller the gauge number
 - 6 gauge: thick battery wire
 - 12 gauge: intermediate wire, used to connect motors

- 16-18 gauge: tiny wire used to connect sensors
- Essential parts of the e-board:
 - Vics: Short for Victors, these are the square black things with fans on top. They are being replaced by the Jaguars (“Jags), the tan things.
 - Like dimmer knobs, Vics and Jags allow variable voltage so the motors can spin at different speeds.
 - Spikes – rectangular boxes with prongs on both ends. Like light switches, they have only two settings: On and Off. They do not have variable voltage. Spikes are used for pneumatics, solenoids, and other components that do not need speed control.
 - Solenoid:
 - Main breaker: fuse panel for the robot; all wires must run through here. 20 or 30 amp fuses are for most components and pneumatics; larger 40 amp fuses are for the Vics or Jags.
 - cRIO: the brain of the robot, it holds control modules for sensors and is where the camera connects.
 - PWM: a bundle wire, consisting of a red, black, and white wire held together by a plug at each end. If the regular wires are the robot’s blood vessels, the PWMs form its nervous system.
- Practical skills
 - In the tool box, do you have...?
 - Wire strippers
 - Wire cutters

- Zip-ties
 - Red and black electrical tape
 - .22 or .32 rosin-core solder
 - Connectors (“terminals”) of appropriate size
 - Safety glasses
- Soldering
 - **Heat the work, not the solder!** Let the soldering iron heat up the work first, **then** apply solder. Don’t just melt the solder onto the work!
- Shrink wrap
 - Rubber tubing used to protect exposed metal connectors
 - Find the appropriate size (slide down wire before putting the terminal on!)
 - Once everything is soldered and cooled, shrink the tubing around the connector using the heat gun (hairdryer on steroids, basically)
- Test board – place to lay out wiring, let programmers try out their programs, insert / take out compartmentalized blocks of electronics as needed
 - Need wires from “brain” to relays for thinking
 - Need wires from batteries to transfer power
 - Need wires to fans for cooling
- Sensors
 - Simple, physical switch – joystick trigger, limit switch
 - Light beam – light is shone in or blocked from a sensor

- Black / white marks on wheels w/ light sensor – measure wheel distance
(x black/white cycles = y feet)
- Shaft/ rotary encoder – rotary angles; degree of turning
- Thermistor (temperature)
- **Ultrasonic transducer** – measure distance, effective up to about 8 feet.
(SUV rear bumper)
 - Mount them on a transducer or pan mechanism – robot can “see” objects around it.
- Infrared – exact distances. No outside interferences. However, can cross up with command signals.
- Microphone
- **Camera**. CMU cam. (Game may have something to do with multicolored lights)
- Accelerometer – acceleration and speed; can be used to indirectly measure distance. Downside: expensive.
- **IR rangefinder** (Sharp GP2 Y0A02YR) Suppliers – acroname; parkfun, etc.
 - Can be connected to anything (?) by input
 - Spot-beam for accuracy
 - Immune to problems of color and reflectivity
 - **Can be used to autonomously drive the robot** – set up sensors on the sides, front, and back. Angled downward, they can sense drop-offs and react accordingly. Robot can do a certain thing

when a predetermined distance range is attained – i.e., can be commanded to drive a certain distance from the wall.

Programming by David E.

TALON 540 has been building robots and volunteering their time to their school, Mills E. Godwin High, and their metro Richmond community for over eight years. Every year, we build a robot for the FIRST (For Inspiration of Science and Technology) Robotics Competition, which has regional events throughout the 50 states and the world. Team 540 competes at the NASA/VCU Regional Competition at the Siegel Center every year, and when we can, attend the National Competition in Atlanta, Georgia.

A big part of building a functional robot is the programming. Without programming, a robot is just motors, pistons, and wheels. The programming is what makes a robot tick: tells the robot what to do, when to do it, and how to do it.

The programs are created on computers using software such as LabView or WindRiver this year. In past years, the programs were created in MPLAB and in the C language. The programs are then deployed, or downloaded, to the “brain” of the robot. This is also different from past years. Before 2009, the FRC Robot Controller (RC) was used, along with the Operator Interface (OI). This year, though, the controls system has changed to the compact RIO (cRIO) and the Driver’s Station. Previously, the signals between drivers, who usually use joysticks to drive and control the robot, and the robot was relayed by radios. This year, a Linksys router is used.

Creating programs in LabView is a different process than MPLAB. C-programming was simple to learn and understand, even if you knew little programming.

For instance, to program a joystick to drive the robot to tank drive, assuming left motor is connected to pwm1 and right is connected to pwm2, it looked a little like this:

```
p1_y = pwm1;
```

```
p2_y = pwm2;
```

p2_y and p1_y correspond to the y axis of joystick 1 and joystick 2. In Labview, though, the program is not so simple-looking.

LabView is an OOP (object oriented programming) language: it's not written in the English syntax of C and there is a "front panel" where you can put lights, gauges, knobs, and other controls and indicators. This makes it much simpler to see the values of variables change and to simulate various conditions without actually testing the robot.

In LabView, variables, such as pwm1 or pwm2, appear as boxes of text on the Block Diagram, which is where programming takes place, and as text boxes on the Front Panel. Instead of the equals sign in C, a line between a "read" variable is connected to a "write" variable: for instance, p1_y would be the write variable and pwm1 would be the read variable.

LabView programming makes much more sense, even to those with little programming experience, once you spend a few minutes looking at the boxes and lines of the Block Diagram and realizing how they correspond to the lights and dials of the Front Panel. We, the programmers of Team 540, have learned a lot of LabView so far and are working on programming the Axis camera to track specified colors. LabView is used a lot in the professional computer sciences and robotics fields and we have enjoyed exploring it and its possibilities.