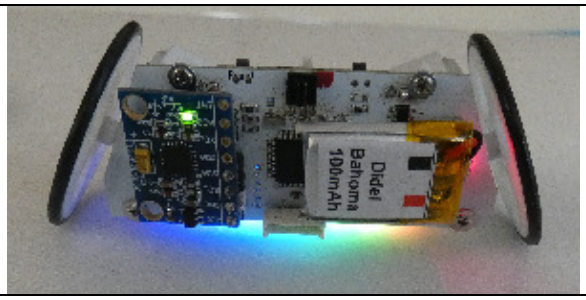
Witty Demo

WittyDemo2.ino (and include files) under www.didel.com/WittyDemo2.zip
The short list of demos is under www.didel.com/WittyQuickStart.pdf
All doc under https://git.boxtec.ch/didel/Witty and www.didel.com/Witty.html

Witty is a new development by DIDEL you should not compare to existing toy's and hacker's robots. It is a piece of technical art, like Swiss watches. Our compact C programs can be easily understood and modified. They run on Arduino environment but have an unusual modular structure.
Witty can also be programmed the Arduino way.



You surely have played with the demos, here are several comments on what you noticed to be unusual, and how to work with our ".h" libraries. We follow the order of the demos and it may be good if you have the WittyDemo2.ino with the .h files on a corner of your screen. See https://www.didel.com/IncludeFiles.pdf if you are not familiar with the ".h" approach.

Program structure
The source is a .ino program developed under Arduino IDE. It uses the setup/loop Arduino structure, but it is plain C for all the programs, no Arduino functions. The functions of the program are distributed in #include files easy to use in any program.
The six tests are accessed via a switch-case structure. The GetPush() function found in GetPush.h counts the pushes after a reset. That function is easy to port on any device having a push-button and a Led. More on www.didel.com/PushButton.pdf (easy link under Witty.html).

1 Simple motor control

The test starts with on-off on one wheel at a time; this shows the linear speed. When both motors starts full speed in the same direction, Witty rotates.
The 4 bits that control the motors are declares by macros AvD; RecD; AvG; RecG; (Avance, Recule, sorry for the French, no applications should use these low-level functions). There are 2 ways to stop the motors, frees running of blocked. The difference is difficult to see, internal amplifier resistance is too high. StopD; StopG; are assigned on the blocked motor macros.
Witty.h lib list the basic operations in addition to the motors: push button, led, infrared receiver module (IRM). The DelMs() function replace the Arduino delay() that load 400 bytes of code, slow down compilation, starts an interrupt we do not wish and, as a function, should have a name starting with an uppercase.
If you really prefer to write `delay()` without Arduino overhead, add the definition:

```
#define delay(x) DelMs(x)
```

2 Proportional control with PFM

See www.didel.com/PFMversusPWMforRobots.pdf to know why we prefer PFM. But PWM is OK (see WittySpecs.pdf).
The demo shows that if the program starts slowly, Witty stays "vertical". Above a given torque on the motors, it starts rotate. This torque depends on the angular velocity, that can be measured with the gyro – interesting to study!
The DoPfm function found in Pwm.h is called by interrupt and the decision to send a pulse of 2ms is taken every 2ms. PFM values are here between -80 to +80.
Note that sei(); is not inside the setup bloc, due to the on/off motor control of demo1. AvG; StopG; etc lasts max 2ms till the DoPfm() function, that uses the same primitives, update the motor lines.

Interrupts, as we are used to have them, interrupt every 58 microseconds. DoPfm is called every 35 interrupts. See www.didel.com/InterruptTimer2.pdf and understand the advantages of having a single interrupt in the system.

3 TV remote have all different formats and decoding is tricky. Using "Morse code" is slow, but it is so simple ! Note the green Led shows the pulses received. If you see them, you just have to click 1,2,3,4 times with the good pulse durations. You can get faster reactions changing the timings. See www.didel.com/TelecommandeIrSimple.pdf

4 The Gy521 accelerometer/gyro is read and its value sent to the terminal output. The port must be selected, but no need to reload the program. Click terminal icon.
Note the Arduino Serial is not used, see https://git.boxtec.ch/didel/TerSer for details.
As explained on www.didel.com/WittyGyro.pdf accelerometer values on y and z can be used to measure "verticality" and experiment with control algorithms. On test 4, simple comparisons on AcZ and AcY change the color when Witty rotates.

5 The Apa102 RGB strip is initialized by the Rainbow of the Apa102.h lib. The PWM values for the light intensity vary between 0 to 255. But we go through a table and progress with 16 levels through the value 0,1,2,4, 6,9,12,17, 27,37,53,78, 100,144,198,255. Being linear is no sense with Leds and little sense with motors. See www.didel.com/Apa102.pdf for the list of available functions, easy to complete.

6 With 8 RGB leds, one can do all kinds of decorative tricks when the Witty is moving. Using the Gy521 to maintain the verticality is of course required.
As a program example with the Apa102 strip, the function ShowByte (v8,rr,gg,bb); count on the strip. The color values rr,gg,bb have a value between 4 to 256 (but more than 40 is indeed too bright). There is a trick for displaying the bits at zero state. If the intensity is a multiple of 4, zero bits are blank. Multiple of 4 plus 1,2,3, the zeros will light slightly with the intensity 1,2,3, this improve the readability.
Mixing colors, programming a rainbow using linear approximations is explained on www.didel.com/RGBstrips.pdf

Jdn 190126/190506