# ATF/GAL blaster project

Goals:

- Build a programmer for GAL devices from Atmel and Lattice
- Build on ATFBlast and GALBlast
- Use an Arduino as computer to avoid reliance on old hardware
    - Connect via serial terminal, so no software on PC needed
    - Upload JED file via ZModem
- ZIF socket that supports multiple ICs

Circuit theory:

The board contains a socket for an Arduino Nano. Different GAL devices use different pins for the programming signal. While the parallel port solution uses different sockets for different GALs in order to connect the right pins, the plan for this programmer is to move this logic to software. However, this means we need much more pins. An Arduino Nano does not have sufficient pins to connect to all of the required pins.

Therefore an MCP23017 I/O expander has been used to add 16 additional GPIO pins, which connects to the Arduino Nano via I2C. An earlier version of the design used 74HC595 shift registers for this purpose, which are output only. I tried to connect all lines that only required output to the shift registers and lines that also might have required input directly to the Arduino, but later I realised that the RAx lines also need input capability should you want to read a configuration from a GAL. Therefore I switched to the MCP23017, but the pin assignment has been kept.

The board contains a ZIF socket and, depending on the type of GAL, Vcc and GND are on different pins. Most GALs have Vcc on pin 28, but the GAL26V12 needs it on pin 7. Transistors Q4 and Q5 allow the Arduino to switch on/off the Vcc on these pins.

Likewise, GND is needed on pin 9, 12 or 21. The Arduino can route this pins to ground by means for MOSFETS Q1, Q2 and Q3.

The Arduino Nano is powered via USB and delivers 5V to the board.

Vpp is generated from 5V by means of an XL6009 boost converter. In order to make the voltage software controllable, the voltage divider on the FB pin, consist of transistor Q6 on the upstream and a normal resistor R3 downstream. The transistor thus acts as a variable resistor. Arduino pin D9 can do 16 bit PWM, which is converted into an analog voltage by capacitor C3. Therefore, by modifying the PWM duty cycle, the Arduino can control the Vpp voltage. The rest of the Vpp circuit matches the application circuit from the XL6009 datasheet. The EN pin is connected to an Arduino pin, which means the Arduino can enable the Vpp voltage by software.

Led1 and led2 are two software controllable indicator leds which can be used as power led or programming led (to be determined at a later stage).