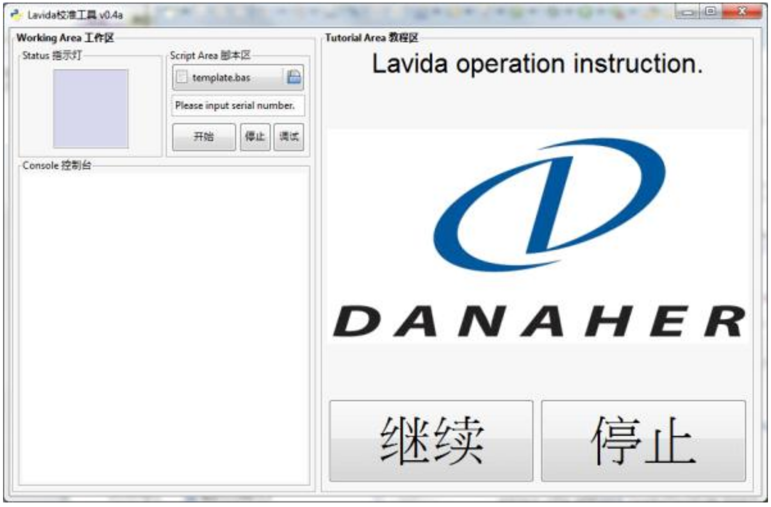
LFT Datasheet v0.4a

2013年1月15日
10:06

- **Environment**:
    - Python-2.7.3
    - 3rd party library: Gtk-2.24.2, pyserial-2.6, ply-3.4, python-msp430-tool, libMSP430-3.2.5.4, cairo, chardet-2.1.1, pexpect-2.3
    - Development Tools: Eclipse(PyDev)/Glade-3.8.1
    - Supported OS: Raspberry Pi - Debian/ Win7 (Only tested OS was list here)
- **Features**
    - Support standard input/output over UART protocol on selected port.
    - Support script interpretation and execution.
        i. With *.cali script, the program will execute the command line by line.
        ii. With *.bas script, the program has realized a BASIC parser using Lex&Yacc algorithm. (Check template.bas for detail)
    - Support automatically and manually result checking function.
    - Support drawing certification document via printer (local printer, network printer or PDF file, depends on OS driver).
    - Support certification document backup with a *.PDF file and test data log with a *.LOG file.
    - Support MSP430 flashing operation via Ti's FET430UIF (JTAG).
    - Support multiple threading function to make sure the GUI is responsiveness all the time.
        i. Thread GTK is used for GUI maintains.
        ii. Thread Receiving is used for receiving text from assigned UART port.
        iii. Thread Plying is used for script interpretation.
        iv. Thread Batching is used for script/command execution.

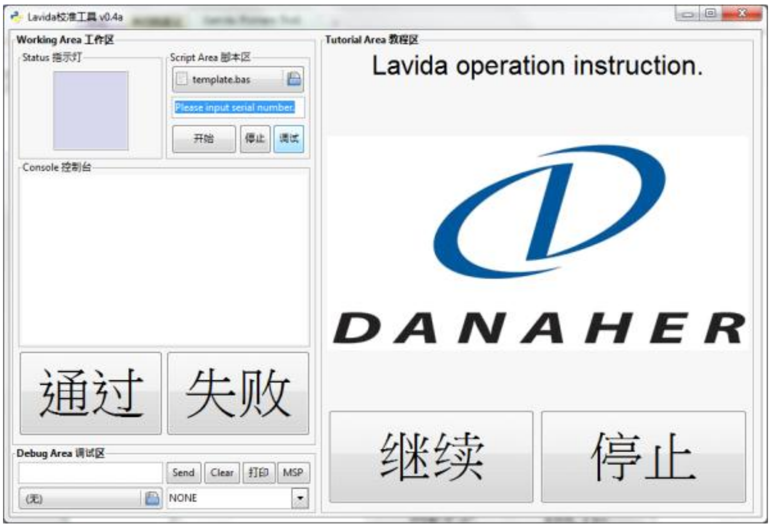
- **GUI demonstration**
    - Main window
        i. Normal operation GUI.



Please input serial number.

Lavida operation instruction.

        ii. Advanced operation GUI.



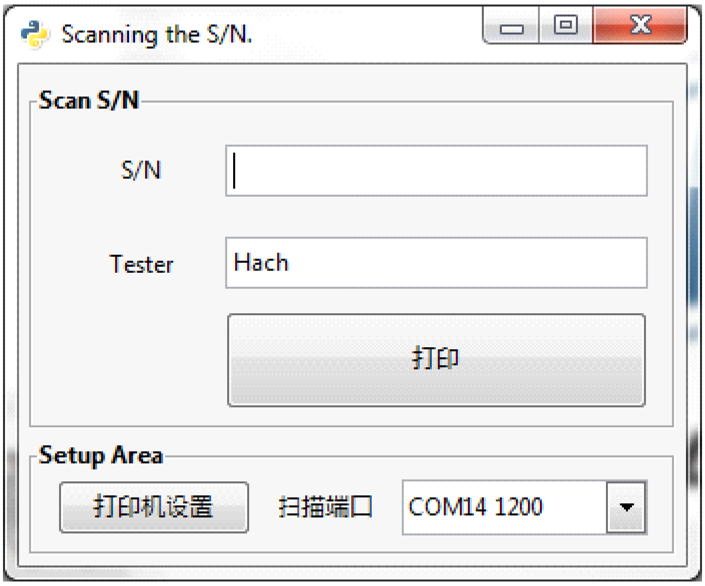
Please input serial number.

(无)

NONE

Lavida operation instruction.

iii. Bar code scan window

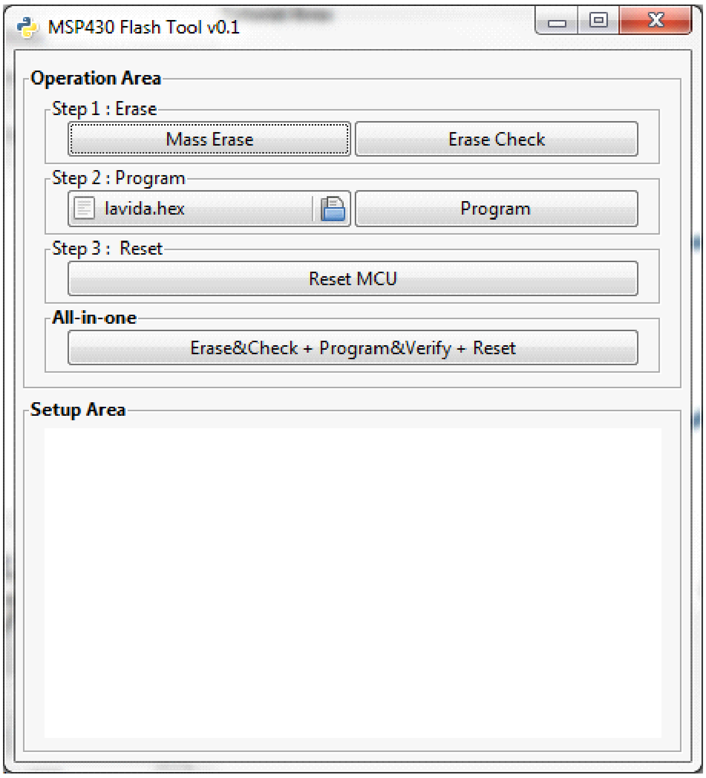


S/N

Tester: Hach

扫描端口: COM14 1200

iv. MSP430 download window



lavida.hex

LFT User Guide v0.4a

2013年3月8日
9:07

Script programming instruction.

We have implemented a standard ®BASIC environment to fulfill the requirement of complex flow control in mass production. In this document, we only list the customized commands, for the exclude standard commands, like FOR NEXT, IF THEN, LET, etc. Please find in ®BASIC's manual at www.cs.bris.ac.uk/~dave/basic.pdf

1. LETSTR is used for assigning string value to a variable, for example:
   a. 100 LETSTR A = "User Guide"
      Will assign string "User Guide" to variable A.
   b. 100 LETSTR B = A, " Ver 1", "(20130308)"
      Variable B will be combined as string "User Guide Ver 1(20130308)"
2. OUT is used for sending commands to connected UARTs, or console that can be used to control instruction label, tutorial image, even sending command to main window. Before using out command, you need to do some initialization:
   a. Initialization for command OUT.
      ```
      40 REM ======== DO NOT CHANGE BELOW VARIABLE NAME ========
      41 LETSTR TUTORIAL = "TUTORIALs"
      42 LETSTR INSTRUCTION = "INSTRUCTIONs"
      43 LETSTR FLASH = "FLASHs"
      44 LETSTR CONSOLE = "CONSOLEs"
      45 LETSTR MCUPORT = "COM2"
      46 LET MCUBAUD = 9600
      47 LETSTR DEVICEPORT = "COM3"
      48 LET DEVICEBAUD = 9600
      50 REM ======== DO NOT CHANGE ABOVE VARIABLE NAME ========
      ```
   b. Sending commands to the specified UART port.
      i. 120 OUT MCUPORT = "10k"
         Will send command "10k" to COM2
      ii. 130 OUT DEVICEPORT = "?NTC.OHM0"
         Will send command "?NTC.OHM0" to COM3

      On Linux, the UART port is represented as #TODO#
   c. Sending commands to instruction label.
      140 OUT INSTRUCTION = "Welcome to Lavida's mass production line. \n欢迎来到 朗逸 生产线！ ", "\nFONTCOLOR=BLUE", "\nFONTSIZE=20"
      Will output below formatted text to the instruction label on main window:"

      Welcome to Lavida's mass production line.
      欢迎来到 朗逸 生产线！

      The "\n" inside the text is used for line break. And the FONTCOLOR and FONTSIZE are used for controlling text's color and size. Be sure to keep the leading character "\n"
   d. Modify tutorial image.
      142 OUT TUTORIAL = "favorite.jpg"
      Will change the picture of tutorial label. Make sure the image is stored in folder "/calibration/tutorials".
   e. Programming MSP430.
      144 OUT FLASH = "./ihexfiles/lavida_ph0121_1.hex"
      Will burn the image to MSP430's flash. In default, the program will do the operation of erase, erase check, program, verify, and reset the chip. Make sure the image *.hex is Intel hex format and stored in folder "/calibration/ihexfiles/"
   f. Sending commands to console
      i. 150 OUT CONSOLE = "Any text you want to put onto the console window"
         Will put the text "Any text you want to put onto the console window" onto the console window.
      ii. 151 OUT CONSOLE = "_CLEAR"
         Will clear the console window.
      iii. 152 OUT CONSOLE = "_CURRENTTIME"
         Will put the current date and time onto the console window.
      iv. 153 OUT CONSOLE = "_ERROR"
         Will send error signal to main window, making status LED to RED, and then terminate the script by program itself.
      v. 154 OUT CONSOLE = "_SCAN"
         Will invoke barcode scanning window.

    vi. 155 OUT CONSOLE = "_MSP430"
        Will invoke MSP430 flash programming window.

3. READDATA is used for reading returned value generated by command "OUT". e.g.:
   160 OUT DEVICEPORT = "?NTC.OHM0"
   162 READDATA ABB = "?NTC.OHM0"
   Will read the result returned from command "?NTC.OHM0" to variable ABB. The READDATA is a READ-CLEAR type command, once the result is read, it will be removed from main program's memory.
4. DELAY seconds.
   170 DELAY 1.1
   Will hold on for 1.1s.
   172 DELAY 0
   Will hold on forever unless human pressed the button "Pass" or "Stop" on main window.
5. STOP command is executed once human pressed the button "Stop" on main window. This design is used for clean-up job before the script is terminated. For example:
   100 DELAY 0
   - ➔ Once user press “fail or stop”,
     - ■ If the STOP exists, the pc will point to STOP
     - ■ If the STOP command is not provided, the pc will point to the END

   900 STOP
   910 OUT CONSOLE = “FAILED OR STOPPED”
   920 OUT MCUPORT = “SHUTDOWN”
   1000 END

   - ➔ Once user terminated the script at any time, the behavior is same as above
   - ➔ Only one STOP command should have because I only accept the first STOP command I have found.

LFT TODO List

2013年3月14日
13:44

- **TODO List**:
  - Add interface for choosing the programming target (all/main/information) of MSP430 flash.
  - Support multiple UART operation via BASIC script. (Currently only supporting 2 UART ports.)
  - Add a short-cut to MSP430 programming window onto Pi's desktop.
  - Add a small desktop tool to adjust Pi's system time and write back to external RTC.
  - Make the name on GUI and in script unified. (e.g.

    100 OUT TUTORIAL = "Image.png" #it's better to change the TUTORIAL to TUTORIAL_IMAGE
    200 OUT INSTRUCTION = "Please check the PCB" #it's better to change the INSTRUCTION to TUTORIAL_INSTRUCTION
  - Support BASIC script interpreting even the UART is not connected.
  - Integrate the script editor into current GUI.
  - Add a checklist for each operation. (something like

    100 OUT CONSOLE = "CHECKLIST:1:Voltage Calibration"
    101 - 150 #doing voltage calibration
    At the end of voltage calibration test, there is a CHECKLIST on GUI to indicate the status.
    160 #decide go on or terminate the test.

    200 OUT CONSOLE = "CHECKLIST:2:Current Calibration"
    201 - 250 #doing current calibration
    At the end of current test, there is a CHECKLIST on GUI to indicate the status by image.
    260 #decide go on or terminate the test.

    900 STOP
    901 - 950 #tidy up.
    1000 END
  - Support template printing. (e.g. via Excel template)
  - Separate the PASS/NG log file by file name
  - Is that necessary to log console window in real-time? (in case the calitool or the Pi is dead)
  - Is that necessary to port to a touchable LCD?
  - Is that necessary to support USB hot plug/unplug function?