

Пользовательский интерфейс с использованием библиотеки ncurses для EMC.

г. Ижевск

6 мая 2011 года.

Изменение 1.01 от 16 мая 2011 года.

1. Задача — написать интерфейс для использования его в программе управления многоосевым станком. Программа управления называется EMC (сайт с описанием — <http://www.linuxcnc.org>).
2. Интерфейс должен работать с использованием стандартной библиотеки Incurses, lmenu. Написан на ANSI Си.
3. Первоначальный вид интерфейса **keystick** (см. таблицу) нужно перевести в подобие интерфейса «Балтсистем» для станков с ЧПУ.

```

F1 Estop On/Off   F5 MDI Mode      F9 Spndl Fwd/Off  ESC Aborts Actions
F2 Machine On/Off F6 Reset Interp  F10 Spndl Rev/Off TAB Selects Params
F3 Manual Mode    F7 Mist On/Off   F11 Spndl Decrease END Quits Display
F4 Auto Mode      F8 Flood On/Off  F12 Spndl Increase ? Toggles Help

ON
Override: 100%
Tool: 0
Offset: 0.0000

MANUAL
LUBE ON
LUBE OK

SPINDLE STOPPED
BRAKE ON
MIST OFF
FLOOD OFF

--Z HOMED
Z SELECTED

Speed: 60.0
Incr: continuous

Relative Act Pos:
--X--      --Y--      --Z--
0.0000    0.0000    -0.0000
0.0000    0.0000    0.0000
    
```

Это то, что сейчас представляет собой интерфейс **keystick**. Это простейший интерфейс, встроенный в emc который использует ncurses.



Это то, во что нужно перевести интерфейс **keystick** и назвать его **baltsys**. Здесь приведена фотография с дисплея системы NC-201 производства «Балтсистем». Для вызова этого интерфейса в секции инициализации переменной DISPLAY задается имя **baltsys**.

4. При написании интерфейса нужно предусмотреть возможность настройки его под разные конфигурации. Для этого как в интерфейсе NC-201 заложить избыточные окна и клавиши. А назначать им название и переменные уже при настройке под конкретную задачу. Изменения при этом нужно будет вносить в исходный текст (файл **baltsys.cc** в директории «**emc2-sev/src/emc/usr_intf**»).
5. Для подготовки к программированию нужно:
 - Установить Ubuntu 10.04 с поддержкой ядра реального времени (RTAI — 122). Для этого скачивается образ диска с <http://www.linuxcnc.org/lucid/ubuntu-10.04-linuxcnc1-i386.iso>. И полностью устанавливаем Ubuntu 10.04 на жесткий диск.
 - Почитать в <http://wiki.linuxcnc.org/cgi-bin/emcinfo.pl?git> как установить исходные тексты. Вкратце — поставить git, командой «**sudo aptitude install git-core**». Затем получить исходные тексты командой «**git clone git://git.linuxcnc.org/git/emc2.git emc2-dev**». Этой командой создастся директорий emc2-dev с исходными текстами.
 - Установка из исходных текстов описана здесь: <http://wiki.linuxcnc.org/cgi-bin/emcinfo.pl?Installing EMC2>. Особенно нужно выполнить команду «**sudo aptitude build-dep emc2**» — этой командой найдется все зависимости для компиляции emc2. Далее командой «**sudo aptitude install build-essential autotools-dev**» поставится Си компилятор с заголовочными файлами и make утилита, если они еще не стоят.
 - Компиляция emc2 выполняется следующими командами:
 - ✓ **cd src** (вы должны находится в директории emc2-dev)
 - ✓ **./autogen.sh**
 - ✓ **./configure**
 - ✓ **make**
 - ✓ **make install-menus**
 - ✓ **sudo make setuid**

- После этого выполняется команда «. *scripts/emc-environment*» для формирования переменных окружения. Если эту команду не выполнить, то запустится не скомпилированный емс а из директория /usr/bin. И запуск самого емс просто командой «*емс*».
6. При написании программы нужно учитывать, что Xwindows на рабочей станции (пульт управления станком) не будет. Linux будет загружаться до level2 или level3 без поддержки Xwindows. Поэтому в качестве файлового менеджера использовать mc.
7. Перечень изменений которые нужно внести в исходные тексты для ввода нового интерфейса:
- в файл «*emc2-dev/scripts/emc.in*» по аналогии с *keystick* точно так же добавить *baltsys*.

В этом файле правится функция Cleanup() и запуск дисплея в foreground.

```
#####
# 3.2. define the cleanup function
#
# this cleanup function doesn't know or care what was actually
# loaded - it simply kills _any_ processes in it's list of emc
# components
#####
function Cleanup() {

    echo "Shutting down and cleaning up EMC2..."
    # Kill displays first - that should cause an orderly
    # shutdown of the rest of emc2
    for KILL_TASK in xemc yemc emcpanel keystick baltsys iosh emcsh emcrsh emctop mdi debuglevel; do
        if $PIDOF $KILL_TASK >>$DEBUG_FILE ; then
            KillTaskWithTimeout
        fi
    done

    if program_available axis-remote ; then
        if [ ! -z "$DISPLAY" ]; then
            axis-remote --ping && axis-remote --quit
        fi
    fi

    if [ "$1" = "other" ]; then
        echo -n "Waiting for other session to finish exiting..."
        WAIT=$KILL_TIMEOUT
        while [ $WAIT -gt 1 ]; do
            if ! [ -f $LOCKFILE ]; then
                echo " Ok"
                return 0
            fi
            WAIT=$((WAIT-1))
            sleep .1
        done
        echo "lockfile still not removed"
    fi

    SHUTDOWN=`$INIVAR -ini "$INIFILE" -var SHUTDOWN -sec HAL 2> /dev/null`
    if [ -n "$SHUTDOWN" ]; then
        echo "Running HAL shutdown script"
        $HALCMD -f $SHUTDOWN
    fi

    # now kill all the other user space components of emc
    for KILL_TASK in emcsrv milltask; do
        if $PIDOF $KILL_TASK >>$DEBUG_FILE ; then
            KillTaskWithTimeout
        fi
    done

    echo "Stopping realtime threads" >> $DEBUG_FILE
    $HALCMD stop
    echo "Unloading hal components" >> $DEBUG_FILE
    $HALCMD unload all

    for i in `seq 10`; do
        # (the one component is the halcmd itself)
```

```

    if [ `SHALCMD list comp | wc -w` = 1 ]; then break; fi
    sleep .2
done

echo "Removing HAL_LIB, RTAPI, and Real Time OS modules" >>$PRINT_FILE
$REALTIME stop

echo "Removing NML shared memory segments" >> $PRINT_FILE
while read b x t x x x x x m x; do
    case $b$t in
        BSHMEM) ipcrm -M $m 2>/dev/null;;
    esac
done < $NMLFILE

# remove lock file
if [ -f $LOCKFILE ] ; then
    rm $LOCKFILE
fi

echo "Cleanup done"
}

```

4.3.10. Run display in foreground

```

echo "Starting EMC2 DISPLAY program: $EMCDISPLAY" >>$PRINT_FILE
result=0
case $EMCDISPLAY in
    tkemc|mini)
        # tkemc and mini are in the tcl directory, not the bin directory
        if [ ! -x $EMC2_TCL_DIR/$EMCDISPLAY.tcl ] ; then
            echo "Can't execute DISPLAY program $EMC2_TCL_DIR/$EMCDISPLAY.tcl $EMCDISPLAYARGS"
            Cleanup
            exit 1
        fi
        $EMC2_TCL_DIR/$EMCDISPLAY.tcl -ini "$INIFILE" $EMCDISPLAYARGS
        result=$?
        ;;
    dummy)
        # dummy display just waits for <ENTER>
        echo "DUMMY DISPLAY MODULE, press <ENTER> to continue."
        read foo;
        ;;
    keystick)
        if ! program_available keystick ; then
            echo "Can't execute DISPLAY program $EMCDISPLAY $EMCDISPLAYARGS $EXTRA_ARGS"
            Cleanup
            exit 1
        fi
        if [ ! -z "$DISPLAY" ] ; then
            xterm -xrm 'XTerm*metaSendsEscape:false' -ls -e keystick -ini "$INIFILE"
        else
            keystick -ini "$INIFILE"
        fi
        result=$?
        ;;
    baltsys)
        if ! program_available baltsys ; then
            echo "Can't execute DISPLAY program $EMCDISPLAY $EMCDISPLAYARGS $EXTRA_ARGS"
            Cleanup
            exit 1
        fi
        if [ ! -z "$DISPLAY" ] ; then
            xterm -xrm 'XTerm*metaSendsEscape:false' -ls -e baltsys -ini "$INIFILE"
        else
            baltsys -ini "$INIFILE"
        fi
        result=$?
    esac
done

```

```

..
11

*)
# all other displays are assumed to be commands on the PATH
if ! program_available $EMCDISPLAY; then
    echo "Can't execute DISPLAY program $EMCDISPLAY $EMCDISPLAYARGS $EXTRA_ARGS"
    Cleanup
    exit 1
fi
$EMCDISPLAY -ini "$INIFILE" $EMCDISPLAYARGS $EXTRA_ARGS
result=$?
;;
esac

# the display won't return until you shut it down,
# so when you get here it's time to clean up
Cleanup

exit $result

```

- в файл «**emc2-dev/src/emc/usr_intf/Submakefile**» по аналогии с **keystick** добавить **baltsys**.

Там где упоминаются ncurses и keystick добавляем:

```

ifeq "$(HAVE_NCurses)" "yes"
KEYSTICKSRCS := emc/usr_intf/keystick.cc
BALTSYSSRCS := emc/usr_intf/baltsys.cc
endif

```

Там где описаны файлы интерфейсов:

```

USERSRCS += $(EMCSHSRCS) $(EMCRSHSRCS) $(EMCSCHEDSRCS) $(EMCLCDSRCS) $(USRMOTSRCS)
$(HALUISRCS) $(KEYSTICKSRCS) $(BALTSYSSRCS)

```

```

ifeq "$(HAVE_NCurses)" "yes"
USERSRCS += $(KEYSTICKSRCS)
USERSRCS += $(BALTSYS)
endif

```

И где описано как компилировать цели:

```

ifeq "$(HAVE_NCurses)" "yes"
../bin/keystick: $(call TOOBJS, $(KEYSTICKSRCS)) ../lib/libemc.a ../lib/libnml.so.0 ../lib/libemcini.so.0
    $(ECHO) Linking $(notdir $@)
    $(CXX) $(LDFLAGS) -o $@ $(ULFLAGS) $^ $(KEYSTICKLIBS)
TARGETS += ../bin/keystick
../bin/baltsys: $(call TOOBJS, $(BALTSYSSRCS)) ../lib/libemc.a ../lib/libnml.so.0 ../lib/libemcini.so.0
    $(ECHO) Linking $(notdir $@)
    $(CXX) $(LDFLAGS) -o $@ $(ULFLAGS) $^ $(BALTSYSLIBS)
TARGETS += ../bin/baltsys
endif

```

- в файл «**emc2-dev/src/Makefile.inc.in**» по аналогии с **keystick** добавить **baltsys**.

Там где упоминается keystick аналогично описать baltsys

```

# ncurses support for keystick
KEYSTICKLIBS = @NCURSES_LIBS@
HAVE_NCURSES = @HAVE_NCURSES@

# ncurses support for baltsys
BALTSYSLIBS = @NCURSES_LIBS@
HAVE_NCURSES = @HAVE_NCURSES@

```

- и скопировать исходный файл keystick.cc в baltsys.cc в директории «**emc2-dev/src/emc/usr_intf**»
8. Весь интерфейс находится в одном файле — **baltsys.cc** в директории «**emc2-sev/src/emc/src/usr_intf**». Этот файл в оригинале является файлом **keystick.cc**. Править только его.
9. **ЧТО нужно сделать?**
- При вызове окна *progwin* появляется текст загруженной программы. Нужно этот текст вставить в статическую панель основного окна *stdscr* и открывание отдельного окна (клавишей «?» убрать).
 - В режиме AUTO для загрузки исполняемой программы в G-кодах нужно нажать клавишу «O». После

этого в нижней строке появляется путь к G программам и оператор должен сам ввести имя файла. Это заменить и при нажатии клавиши «О» должно появляться меню с перечнем файлов в директории с G программами. Оператор только производит выбор стрелками и нажимает ENTER. Вводить самому имя программы не нужно.

- При вызове окна toolwin (нажатие клавиши «L») появляется статическая таблица инструментов. В этом виде править ее нельзя. Причем если инструмент не определен, то его тоже видно но с номером «-1». Нужно во-первых убрать отображение неопределенного инструмента и выводить только те инструменты, которые заданы. Далее, вызов отдельного окна toolwin убрать и все оформить в окне stdscr. При вызове инструментов появляется форма, где можно изменять все поля, которые задают и описывают инструмент. Переход между полями стрелки или табуляция. Так как станок работает в двух режимах — lathe (токарный) и mill (фрезерный) и таблица инструментов имеет разное число полей. Это тоже учесть при написании формы.
- Четко определить активное в данное время окно и принимать только ввод с клавиатуры который относится к данному окну. К примеру если правится инструмент и вызвана форма с инструментом то от нажатия стрелок не будут двигаться оси (режим «Jog»). Аналогично и с окном diagwin. При его вызове работают только те клавиши, которые участвуют непосредственно в управлении этого окна.
- Все аварийные сообщения выводить не в поток stderr а в отдельное окно. Так как при работе не в графическом режиме при появлении сообщения все окно уплывает и его нужно перерисовывать нажатием комбинации клавиш «Ctrl+L».
- Согласовать расположение информации на экране, положение MDI строки для ввода команды, расположение панелей и т.д. Пока этот раздел находится в разработке (16 мая 2011 года).
- Исключить «засыпание» дисплея при удерживании какой либо клавиши (к примеру управления осями в режиме «Jog»). (Исправлено 15 мая 2011 года).
- Исправить ошибку триггерного непереключения от нажатия клавиатуры и необновления статуса программы. (Исправлено 14 мая 2011 года).

10. Принцип работы интерфейса:

11. Описание исправленных ошибок доставшихся от keystick.

- Исправление непереключения клавиш в триггерном режиме (F1, F2)

<pre>static void idleHandler() { // read NML status updateStatus(); // only print if main is not printing, so we don't clobber in the middle if (!critFlag) { printStatus(); } // simulate key-up event, as per comment below keyup_count -= usecs; if (keyup_count < 0) { keyup_count = 0; /* oldch = 0; */ ch = 0; } }</pre>	<p>В функции idleHandler() анализировалось время keyrepeat. И не той переменной присваивалось значение 0. Раньше было oldch = 0, а правильно нужно ch = 0. Автор исправления Алексей Кудрин (14 мая 2011 года).</p>
---	---

- Исправление «зависания» экрана при удержании клавиш.

<pre>while (!done) { oldch = ch; int keypress = getch(); if(keypress == ERR) { idleHandler(); continue; } ch = (chtype) keypress; // check for ^C that may happen during blocking read if (done) { break; } if (dump)</pre>	<p>В основной функции main() при организации цикла while(!done) если последняя нажатая клавиша равна той, что нажата сейчас (при опросе) не происходил вызов updateStatus(). Соответственно не происходило заполнение изменения данных в stdscr. Изменения в самой программе EMC происходили но не отображались. Поэтому пришлось добавить вызов updateStatus() и printStatus(). Исправлено Алексеем Иноземцевым (15 мая 2011 года).</p>
---	--

<pre> { mvwaddstr(window, wmaxy, wbegx, line_blank); sprintf(scratch_string, "%12o", (int) ch); mvwaddstr(window, wmaxy, wbegx, scratch_string); wmove(window, wmaxy, wbegx); wrefresh(window); if (ch == 'q') break; else continue; } if (ch != oldch) { keyup_count = FIRST_KEYUP_DELAY; } else { keyup_count = NEXT_KEYUP_DELAY; /****** AIKE added *****/ /****** POSITEVLY *****/ /* if not print status, the pressed key value will not displayed immediatly */ // read NML status updateStatus(); printStatus(); } </pre>	
--	--

- Исправление вывода на экран всех инструментов, даже тех, которые не определены. И описание всех полей для токарного станка (7 полей) а не только для фрезерного (4 поля).

<pre> else if (window == toolwin) { wattrset(window, A_BOLD); mvwaddstr(window, 0, 34, "Tool Table"); wattrset(window, 0); mvwaddstr(window, 2, 1, "Pocket ToolNo Length Diameter Orientation Frontangle Backangle"); wattrset(window, A_UNDERLINE); line = 4; for (t = 0; t < CANON_POCKETS_MAX; t++) { if (emcStatus->io.tool.toolTable[t].toolno != -1) /* AIKE */ { sprintf(scratch_string, "%2d%8d%11.4f%10.4f%13d%12.4f %12.4f", t, emcStatus->io.tool.toolTable[t].toolno, emcStatus->io.tool.toolTable[t].offset.tran.z, emcStatus->io.tool.toolTable[t].diameter, emcStatus->io.tool.toolTable[t].orientation, emcStatus->io.tool.toolTable[t].frontangle, emcStatus->io.tool.toolTable[t].backangle); mvwaddstr(window, line++, 3, scratch_string); } } } </pre>	<p>При вызове окна toolwin не проводился анализ на номер инструмента = -1. Добавлен анализ и если инструмент не определен, то он не выводится в окно. И для токарного станка добавлены еще 3 поля. Исправлено Алексеем Иноземцевым (12 мая 2011 года).</p>
--	--