Dr. Bob Davidov

Вычисления в реальном времени с использованием таймера MatLAB

Цель работы: освоение правил построения систем реального с использованием таймера в среде МатЛАБ.

Задача работы: построение систем в которых реальное время отсчитывается таймером МатЛАБ.

Приборы и принадлежности: Персональный компьютер, МатЛАБ, Simulink

ВВЕДЕНИЕ

МатЛАБ имеет средства компиляции модели Simulink для работы в режиме реального времени. Построенная виртуальная система отлично работает самостоятельно и с различными интерфейсами, обеспечивающими связь модели с реальными сигналами внешней среды. Однако компиляция не проходит (и система реального времени не строится) если система содержит неопределенные по времени процессы, например, открытие объектов портов для передачи данных, или обеспечение доступности переменных Workspace во время работы системы. В этой работе рассматриваются средства и варианты построения системы реального времени на базе таймера МатЛАБ, которая имеет связь с GUI и Workspace переменными и может обмениваться данными через каналы COM, UDP, TCP.

ОБЩИЕ СВЕДЕНИЯ

Построение системы реального времени с использованием таймера позволяет достаточно легко построить распределенную систему задействующую самые разные каналы передачи данных: COM, DCOM, UDP, TCP и другие для связи m-файлов, моделей Simulink, Workspace, GUI, физических объектов и программных сред. В таких системах обмен блоками информации может производиться с частотой тактов реального времени. Частота квантования такой распределенной системы ограничена, в основном, максимальной частотой создания объектов и запуска блоков системы. Например, использование модели simulink в указанном режиме реального времени требует выполнение следующей последовательности.

- Установка начальных параметров модели
- Запуск модели
- Чтение и использование промежуточных и конечных состояний модели

- Для обеспечения непрерывной работы модели значения всех интеграторов модели вычисленные к конце последнего такта используются в качестве начальных значений следующего такта.

Следующий пример описывает систему реального времени включающую m-файл, mdl модель и GUI (Graphical User Interface – графический интерфейс пользователя). В примере используется ввод и отображение параметров и обмен данными между указанными блоками системы.

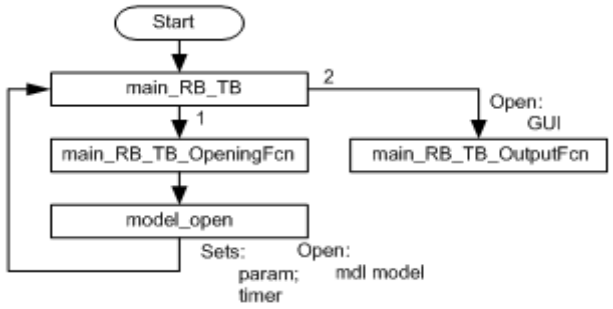

Рис. 1. Запуск программы реального времени с GUI и Simulink моделью через главный модуль main_RB_TB.m

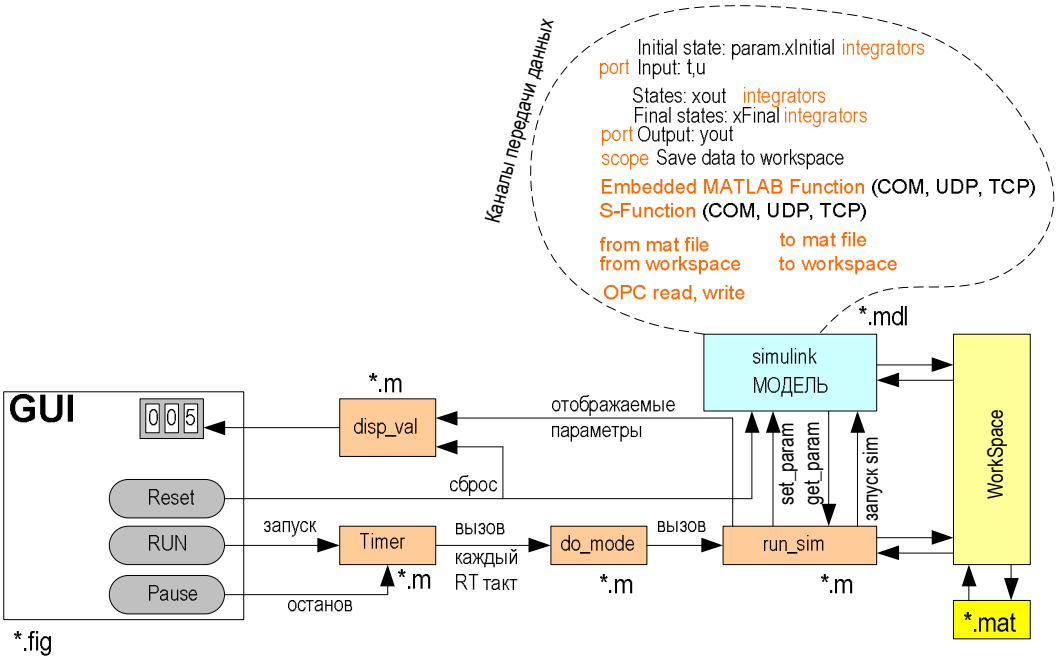

Рис. 2. Взаимодействие программных блоков. Штриховой линией обозначены возможные каналы для взаимодействия с внешними средами.

Команды управления Таймером МатЛАБ.

Команда	Описание
t_array = timerfind и out = timerfindall	Поиск конкретного таймера и всех объектов таймер запущенных ранее

t = timer;
t = timer('TimerFcn',@ mycallback, 'Period', 10.0);
rt_timer = timer('TimerFcn', 'do_mode', 'ExecutionMode', 'FixedRate');
t = timer('StartDelay', 5,'TimerFcn','disp("Hello World!")');
t = timer('TimerFcn', 'my_function');
	Создание объекта "t" таймер.
	Создание объектов "таймер" которые вызывают функции mycallback и do-mode.
	Установка параметров таймера.

rt_timer.Period = 1; set(t, 'ExecutionMode', 'FixedRate') set(t, 'TasksToExecute', 7)	Установка параметров таймера (период = 1 сек.) Примеры установки параметров.
disp и get **>>get(t)** AveragePeriod: NaN BusyMode: 'drop' // [{drop} | queue | error] ErrorFcn: '' // string -or- function handle -or- cell array ExecutionMode: 'singleShot' // [{singleShot} | fixedSpacing | fixedDelay | fixedRate] InstantPeriod: NaN Name: 'timer-1' ObjectVisibility: 'on' //[{on} | off] Period: 1 Running: 'off' StartDelay: 1 StartFcn: '' // string -or- function handle -or- cell array StopFcn: '' // string -or- function handle -or- cell array Tag: '' TasksExecuted: 0 TasksToExecute: Inf // number of periods TimerFcn: '' // string -or- function handle -or- cell array Type: 'timer' UserData: []	Отображение и прием информации объекта таймер Пример выполнения команды.
start(t) и startat(t, 10)	Запуск таймера и запуск таймера в указанное время
stop(t)	Останов таймера
wait	Ожидание окончания работы таймера, например, полного выполнения всего количества заданных тактов
delete(tr_timer);	Удаление объекта tr_timer

Передача структуры handles в подпрограмму do_mode запускаемую таймером осуществляется через глобальную переменную GUIparams.

Функция model_open(handles) устанавливает глобальные переменные

```
global param
```

задает значения структурной переменной

```
param.t_Initial = 0;
```

удаляет таймеры которые находит

```
out = timerfindall;      % Find timer objects used before
delete(out);             % Remove a timer object from memory
```

создает новый таймер и устанавливает его параметры

```
% Create new timer object
```

```
    rt_timer = timer('TimerFcn', 'do_mode', 'ExecutionMode', 'FixedRate');
    rt_timer.Period = 1; % Specifies the delay, in sec, between execut
```
если модель не открыта открывает ее
```
    if  isempty(find_system('Name', 'RB_TB')),
        open_system('RB_TB');
```
устанавливает параметры mdl модели
```
        set_param('RB_TB/Controller/depth_range', 'Value',...
           get(handles.depth_max_edit,'String'))
        set_param('RB_TB/Controller/Saturation', 'UpperLimit',
           get(handles.depth_max_edit, 'String'));
    end
```

Коды программных блоков показанных на рис. 2:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% do_mode.m     v3.1d
% Matlab v7.0 (R14) SP 1
% Bob Davidov
% 23 October 2006
%
% run by timer of main_RB_TB.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function do_mode
global param GUIparams sim_state

handles = GUIparams;  % handles transmitted into do_param via global GUIparams

if param.t_Initial <= 0
   param.xInitial(1:3) = 0; % model mdl contains 3 integrators
   set(handles.Depth,'String','0')
   param.depth_Initial = 0;
   param.yout = [];
end

param = run_sim(param,handles,sim_state); % transmit handles into run_sim

%end of do_mode.m
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% run_sim.m     v3.1e
% Matlab v7.0 (R14) SP 1
% Bob Davidov
% 12 March 2008
%
% run RB_TB.mdl from "param.t_Initial" to "param.t_Final" time
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function param = run_sim(param,handles,sim_state);

t = [param.t_Initial; param.t_Final];

% even or odd clock: 0 or 1. to flash display
clock = round(rem(param.t_Initial/2,1));
delta_time = param.t_Final - param.t_Initial;

%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% run simulink model
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
param.u = param.u + 0.1;
param.t_Final = param.t_Initial + str2num(get_param('RB_TB/RT_clock', 'Value'));

sim('RB_TB', [param.t_Initial, param.t_Final], simset('InitialState', param.xInitial), [t, param.u])

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% save sim output
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
param.t_Initial = param.t_Final;
param.xInitial = xFinal;
param.depth_Initial = param.depth_Final;
param.yout = yout;

%--------------------------
% MatLAB GUI output
%--------------------------
size_yout = size(yout,1);

depth = yout(size_yout,1);          % DEPTH m
NewVal = round(depth);
disp_val(NewVal, 0, 600, 3, 14, handles);
set(handles.LED, 'BackgroundColor', [0 0 (1 - NewVal / str2num(get(handles.depth_max_edit,'String')))]);

NewVal = round(yout(size_yout,2));          % DIVE TIME
time_d = floor(NewVal*100/3600)*100 + NewVal - floor(NewVal/60)*60; % transmit ..s5s4s3s2s1s0 into h0m1m0s1s0
disp_val(time_d, 0, 95959, 5, 19, handles);

% end of run_sim.m
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% disp_val.m     v3.1d
% Matlab v7.0 (R14) SP 1
% Bob Davidov
% 23 October 2006
%
% sends symbols to MatLAB GUI
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function disp_val(value, treshold_min, treshold_max, order, last_symbol, handles)

value = abs(value);
value_code = uint8(num2str(value));
value_str = int2str(value);

for i = 0:order-1
   do_switch(last_symbol-i, '0', handles);
end

s = size(value_str, 2);
for i = 0:s-1
   do_switch(last_symbol-i, value_str(s-i), handles);
end
```

```
function do_switch(symbol_num, symbol, handles)
switch symbol_num
   case 12
      set(handles.symbol12, 'String', symbol)
   case 13
      set(handles.symbol13, 'String', symbol)
   case 14
      set(handles.symbol14, 'String', symbol)
   case 15
      set(handles.symbol15, 'String', symbol)
   case 16
      set(handles.symbol16, 'String', symbol)
   case 17
      set(handles.symbol17, 'String', symbol)
   case 18
      set(handles.symbol18, 'String', symbol)
   case 19
      set(handles.symbol19, 'String', symbol)
   otherwise
      disp('Unknown GUI symbol number.')
end
```

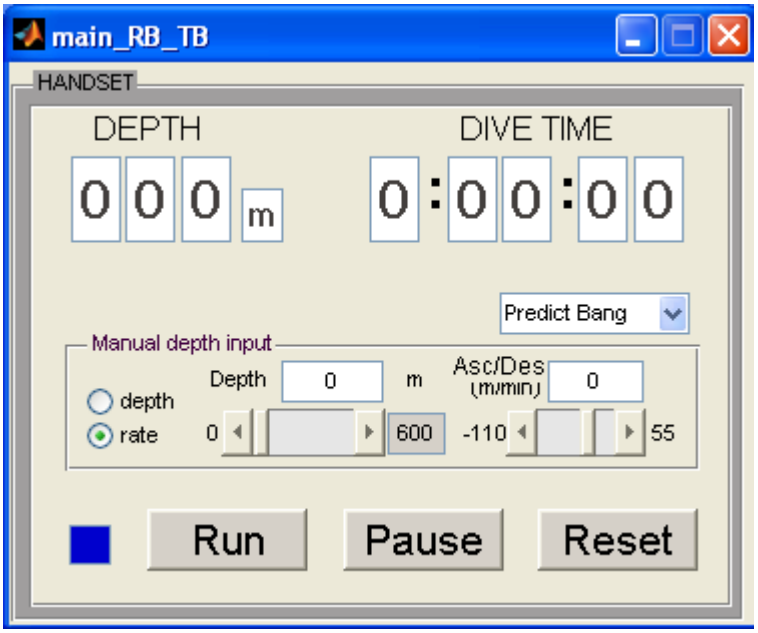

Рис. 3. Графический интерфейс пользователя GUI

GUI (Graphical User Interface -графический интерфейс пользователя) обеспечивает

- отображение параметров Depth и Dive Time
- выбор режима из списка
- переключение режима
- ввод параметров через клавиатуру и движением мыши
- ограничение параметров
- цветовое сопровождение изменения параметра Depth
- управление состоянием клавишами Run, Pause и Reset.

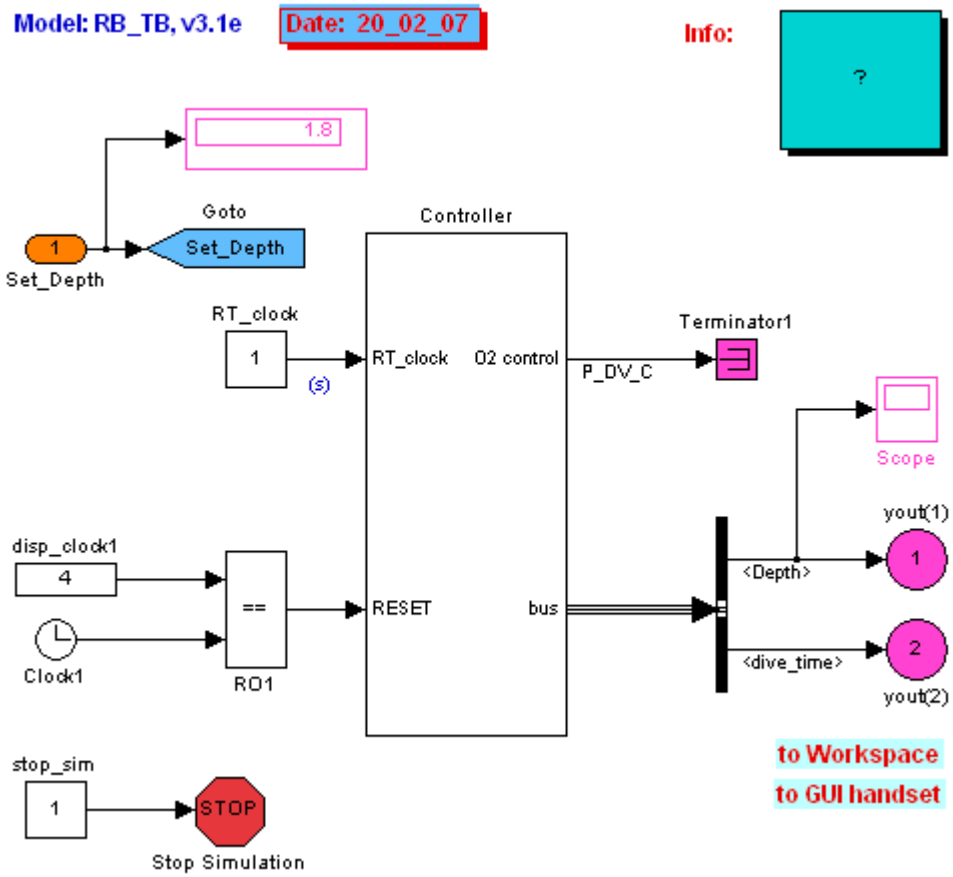

Рис. 4. Основной модуль mdl модели

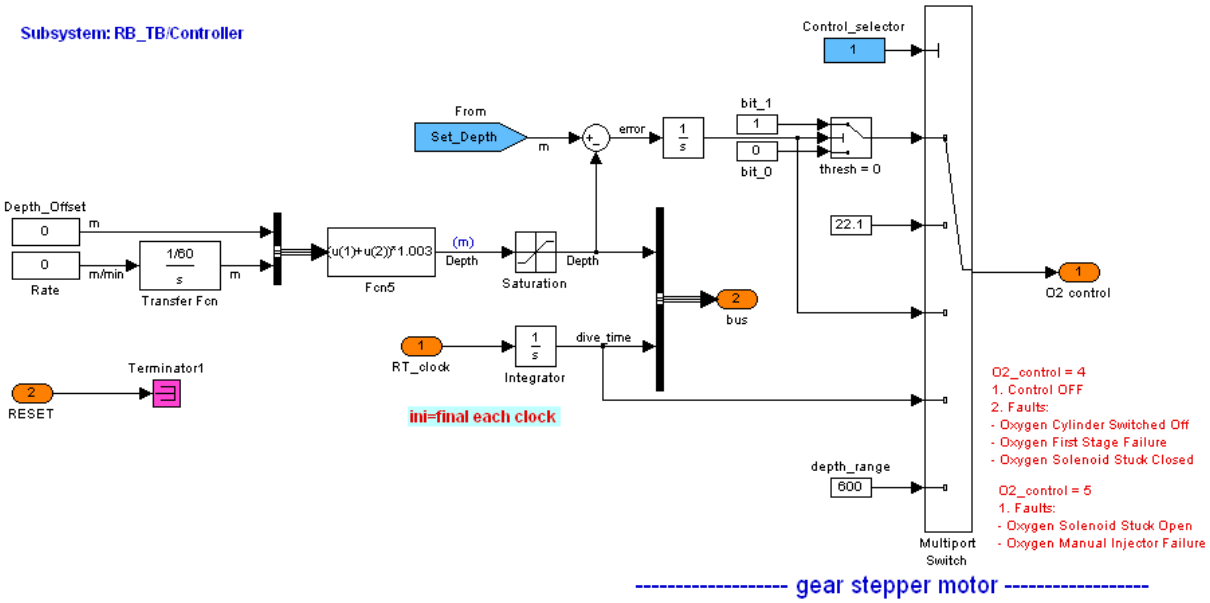

Рис. 5. Подсистема модели“Controller” основного модуля

Структура handles в подпрограмму run_sim которая запускает simulink модель передается в списке входных параметров run_sim. do_mode вызывает run_sim.

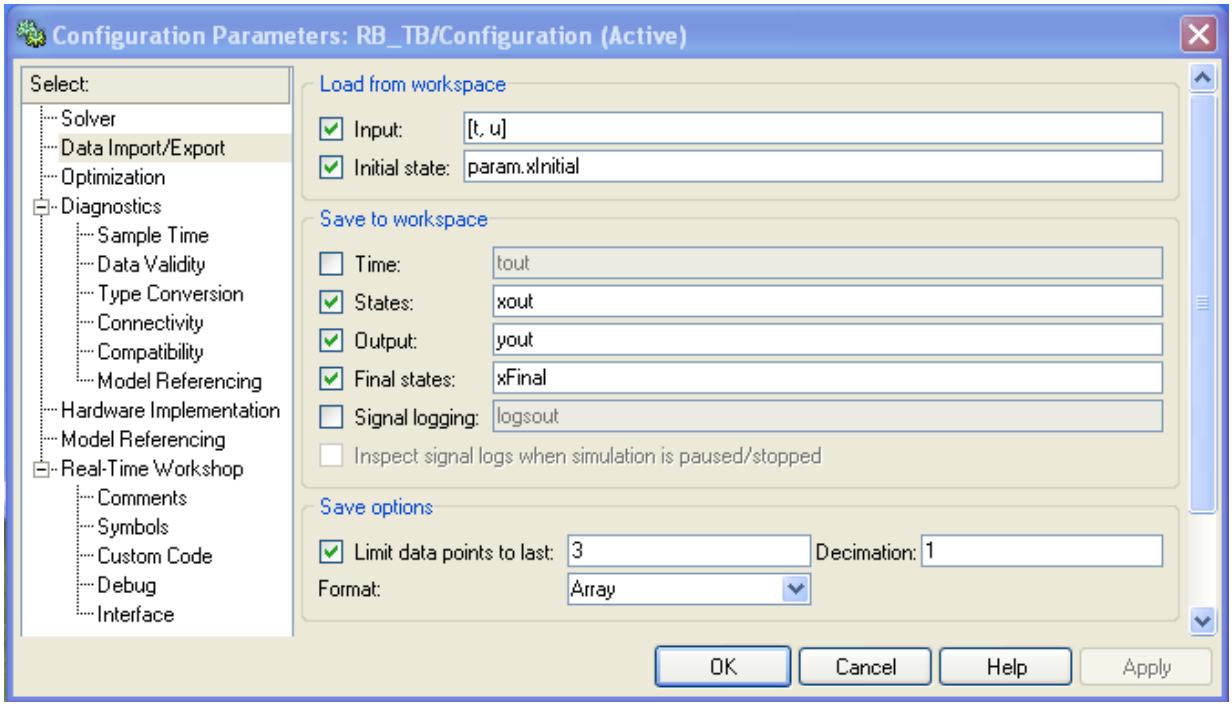

Рис. 6. Параметры моделирования

Каналы связи “модель – workspace” (входные выходные параметры) устанавливаются в разделе меню модели > Simulation > Configuration Parameters (см. рис. выше):

Input: [t, u] - время и значения входного порта (портов). Вектора столбцы. Значения считываются из workspace непосредственно после запуска модели;

Initial state: - начальные значения интеграторов модели (считываются из workspace);

States: Последние значения интеграторов модели которые поступают из модели в workspace после окончания ее работы. Количество таких значений задается параметром “Limit data points to last”. Самое последнее значение присваивается параметру “Final states”

Output – имена выходных портов модели значения которых попадают в workspace по окончанию моделирования

Final states - последние значения интеграторов модели

Limit data points to last ограничивает размер массива “States” и блоков Scope последними полученными значениями.

Примечание. Максимальная длина вектора States: равна отношению времени работы модели (Stop time – Start time) к шагу вычислений: Fixed-step size раздела меню модели > Simulation > Configuration Parameters > Solver.

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Задание 1. Построение модели реального времени с использованием таймера МатЛАБ.

Выполните следующие пункты используя данные примера приведенного выше.

1. Наберите модель simulink (см. рис.4 и рис.5) и установите параметры моделирования (см. рис.6).
2. Командой guide вызовите редактор графического интерфейса и постройте GUI (см. рис.3).
3. Включите таймер в главный m-файл GUI. Таймер должен запускать do_mode.m, который вызывает модуль run_sim.m управляющий запуском модели и отображением состояния модели в окне GUI при помощи модуля disp_val.m (см. параграф “Коды программных блоков”)
4. Используя команды set_param, get_param; set(handles,…), get(handles,…); Simulink Configuration Parameters: Input, Initial State, States, Output, xFinal; и каналы блока Scope обеспечьте динамическую связь между GUI, m-файлами и Simulink моделью.
5. Обеспечьте совместную работу таймера, GUI и mdl модели.
6. Подключите к системе новые объекты или системы используя каналы COM, DCOM, UDP, TCP (см. соответствующие работы).

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какими свойствами обладает система реального времени?
2. Сравните возможности систем реального времени под управлением таймера МатЛАБ и моделей Simulink работающих в режиме реального времени после компиляции.
3. Какая из перечисленных выше систем позволяет обеспечить связь с OPC сервером, UDP каналом?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Help МатЛАБ