Microwave oven control - case study

Usage of StateWORKS development tools

Topic

The control of a microwave oven is relatively simple and does not present any challenge to the designer. We have chosen it to show the usage of real-time data base (RTDB) objects. The RTDB objects are of various types and may realize some control functions. A correct use of these functions simplifies the state machines. Some obvious examples of control functions are counters which count commands, events, clock pulses (timers), etc. In this example we want to show the usage of switch points (Swip), which in RTDB based systems are used to supervise whether an object value stays within limits.

The requirements are (we repeat here the text from StateWORKS page):

The oven has a Run push button to start (apply the power) and a Timer that determines the cooking length. Cooking can be interrupted at any time by opening the oven Door. After closing the Door the cooking is continued. Cooking is terminated when the Timer elapses. When the Door is open a Lamp inside oven is switched on, when the Door is closed the Lamp is off.

The control system has the following inputs:

Run push button - when activated starts cooking,

Timer - while this runs keep on cooking,

Door sensor - can be true (door closed) or false (door open).

And the following outputs:

Power - can be true (power on) or false (power off),

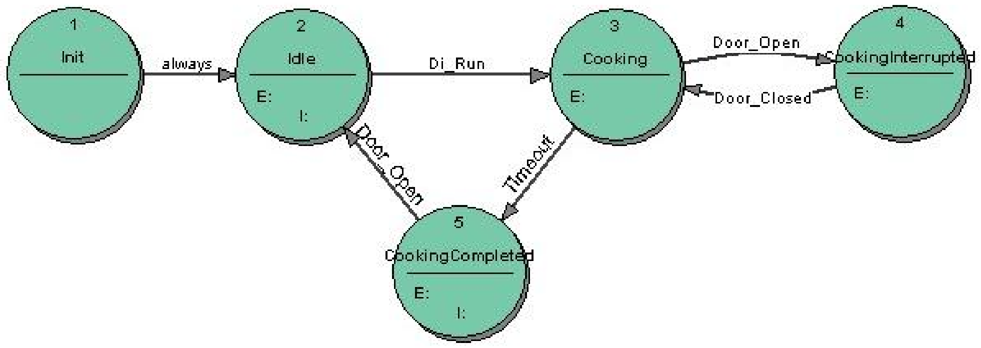
Lamp - can be true (lamp on) or false (lamp off).

The knobs to set the power and timeout values are irrelevant for the control state machine. The behavior of the microwave oven control is determined by the Run push button, Timer and Door sensor.

First simple solution

The first approach is simple and we consider only the basic requirements as written down above. We will later analyze the missing control function and specify a more complete control system.

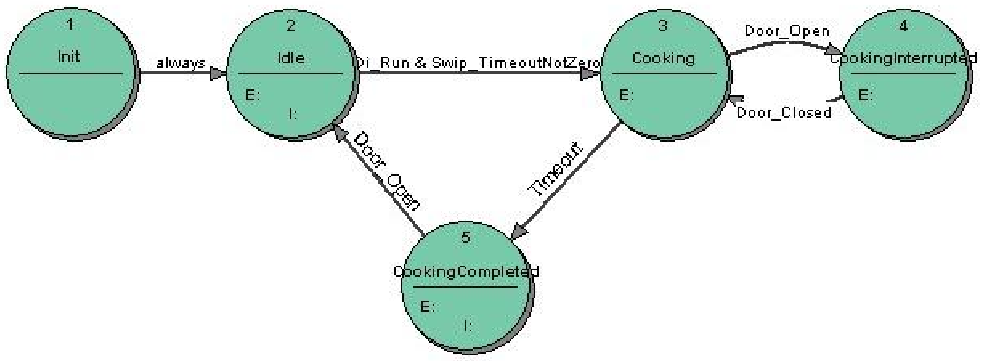
The state transition diagram for the simple solution is shown below.



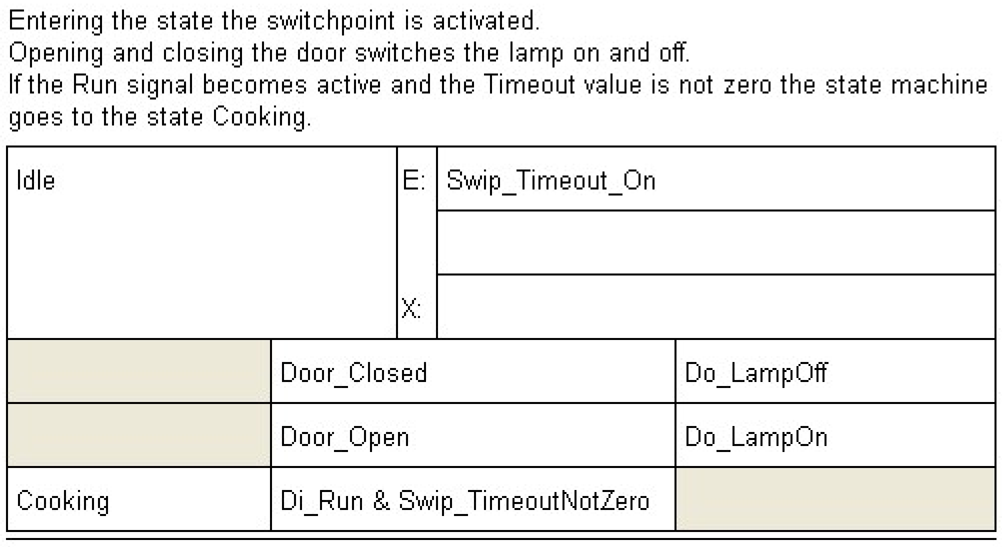
The details of the state transition diagram are in the state transition table. To display it you need to use the program StateWORKS Studio. For the purposes of this study we show here only the table for the Idle state.

This simple solution has some weak points. Of course, microwave ovens are manufactured in different flavors with slightly different controls. Thus, we cannot analyze all possible variants we see on the market. For purpose of our exercise we will consider the timer. The timer usage is not perfect, namely the control system always starts, even if the timeout value is set to 0. To reset the system for the next start we have to open and then close the door: if you do not like this then you will find the exercise of changing the design very educational.

More realistic control



To achieve a more elaborate control shown above we introduce a condition Swip_TimeoutNotZero which together with with Di_Run (using logical AND operation) forms the condition for the transition from the state Idle to Cooking. A Switch-point in the RTDB is a mechanism for testing any numeric value, and producing a set of Control Values in the positive-logic convention employed by StateWORKS to govern possible transitions. Instead of using directly the Control Values we use control names (see more detailed explanation later in “RTDB object” section). Commonly, we employ names such as "In_Permitted_Range", "Too_High" and "Too_Low". Here we only need to use the one "name" Swip_TimeoutNotZero.)

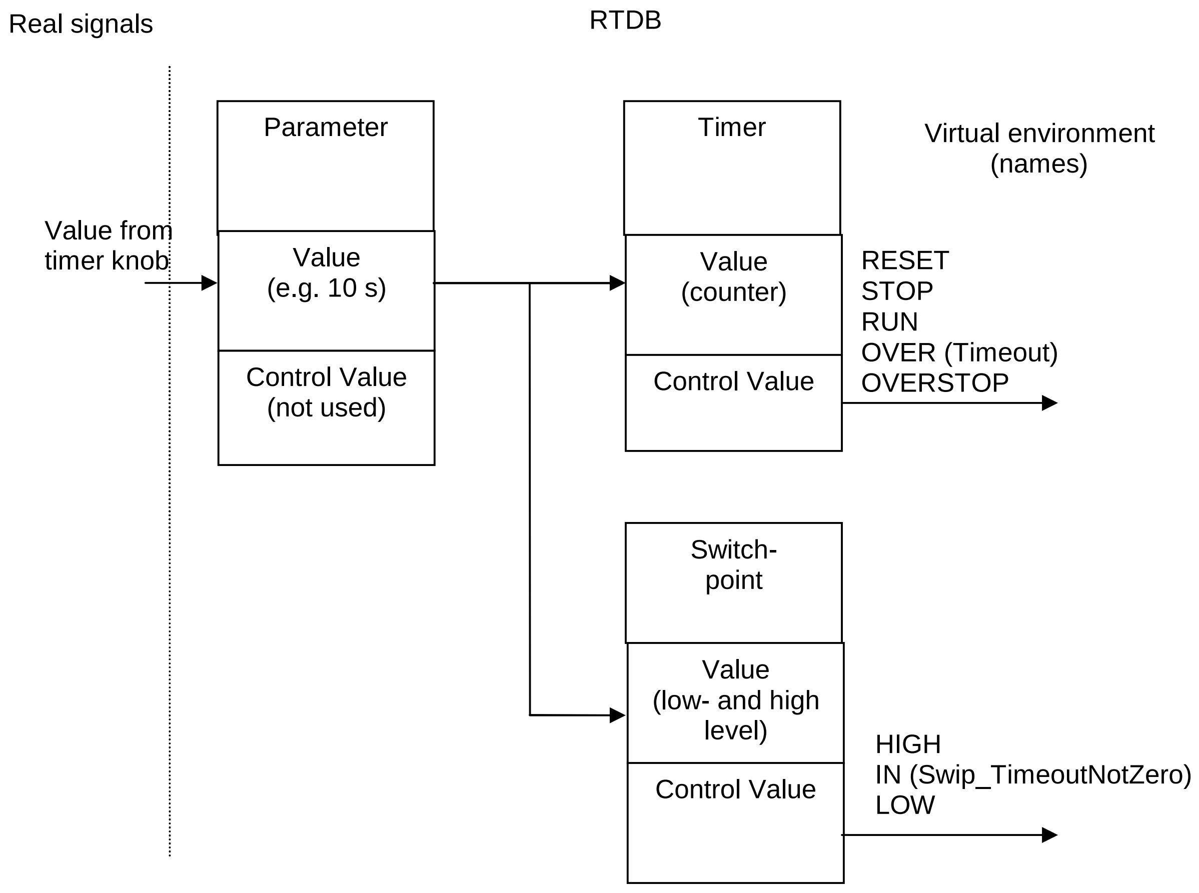


Entering the state the switchpoint is activated.
Opening and closing the door switches the lamp on and off.
If the Run signal becomes active and the Timeout value is not zero the state machine goes to the state Cooking.

Idle	E:	Swip_Timeout_On
	X:	
	Door_Closed	Do_LampOff
	Door_Open	Do_LampOn
Cooking	Di_Run & Swip_TimeoutNotZero	

The specification is abstract, so, we do not care at this moment how we might get the conditions. At this moment, the VFSM specification defines a link to the real object in the RTDB by choosing the needed object types and using their control values to define control names, as for instance Swip_TimeoutNotZero.

Later, during the detailed system configuration, we can decide which specific timer will be used for oven control and link its timeout value with a parameter (set by a timer knob on the microwave oven front panel). That parameter, in turn, will be an object to be supervised by a switch point. We may show the dependencies specified in the RTDB by the following diagram:



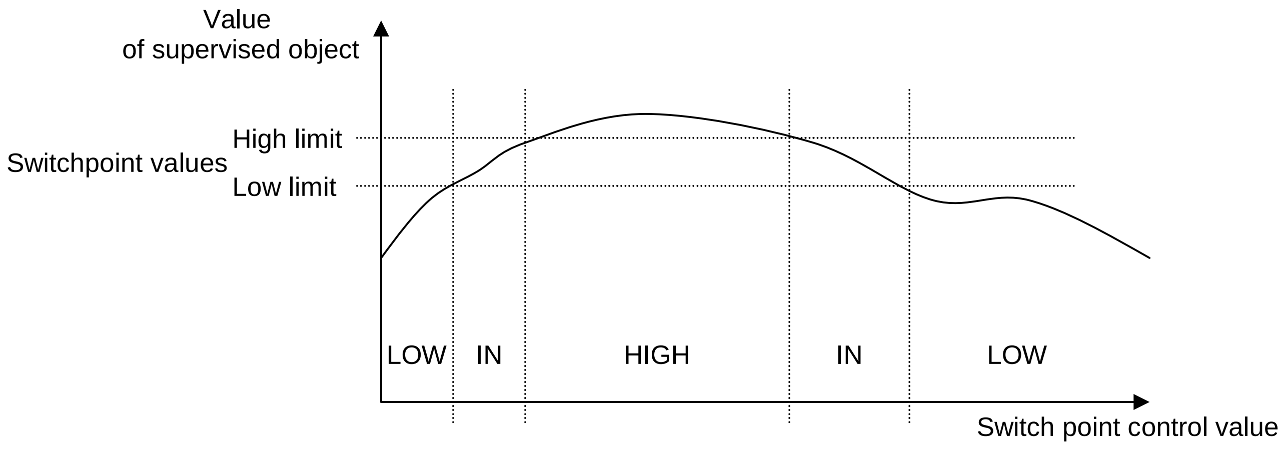
RTDB objects

The RTDB consists of objects. Objects are of different types i.e. they have different properties; specifically they have different Values and Control Values.

A parameter object may have Value of type integer, float, string, etc. It may also have a Control Value but we do not discuss it here as it is irrelevant to the example. In our example Parameter stores the timeout value which is an integer representing the number of seconds.

The timer object value is its counter output which represents the elapsed time from the timer start. The timer object compares continuously the counter output with the timeout value (in this case supplied by the Parameter object) and defines the timer Control Value. If the timer is not started yet the Control Value is RESET. If the timer runs (the counter counts some time pulses) the Control Value is RUN. If the counter output equals the timeout value the Control Value is OVER.

A switchpoint object (SWIP) compares some object Value with its Values which represent in this case Low and High levels according to the following diagram:



In our example the Switchpoint object compares its values (1 and a very large number) with Parameter value which is a timeout. The result of this comparison is Switchpoint Control Value which may be LOW, IN or HIGH. As we are interested only in the value 0 of the Parameter we use the Control Value = IN as a condition name Swip_TimeoutNotZero. Note that the IN range includes the two limit values.

Yet another change

To demonstrate the flexibility of RTDB we will show you how to change the timeout value if it is determined by a potentiometer. The potentiometer delivers an analog signal - a voltage. So, we use a Numerical Input (NI) object in the RTDB to store the voltage and use this object as a source of Const value for the Timer object and as an Input for the Switchpoint object. In other words, the only change is to replace the Parameter with the Numeric Input.

Conclusions

A state machine determines the behavior of a control system. The complexity of the state machine depends, among other factors, on the means which are to be used to build the control system: the input / output system (hardware interface) and the system resources (timers, counters, etc.).

The RTDB provides a set of already-prepared objects which can simplify the design of a control system by implementing some general control functions. In this paper we have shown one of these control functions: supervision of a certain value.

You may have noticed that the requirements as specified in the beginning were not very detailed. We did that on purpose as it is often the situation with which we are confronted in a real project. Not till we see the first solution we “discover” that it is not what we have expected. Well, formally the first simple solution fulfilled the requirements. To start a project with an incomplete specification is not desirable, but it is a common practice and often unavoidable. Anyway, eventually we completed revision of the requirements and the second solution seems to be more realistic.

A real project may require additional control functions. A microwave oven usually has a rotating platform which turns when the power is applied: separate controls for the power and the motor may be required. There are microwave ovens which “store” the Start signal even when the timeout is 0: in such a case we may first push the Start button and later, at any time, we start cooking by setting the timer. Another problem may be the setting of the timeout value: it may be a direct digital signal, an input from a key-board, or an analog setting adjusted by a potentiometer: these possibilities require different solutions in the RTDB. However the details may vary, any requirements can be transformed to a neat VFSM specification implemented by means of the StateWORKS system.

We have used such very simple examples in order to illustrate some of the important aspects of StateWORKS: the usage of RTDB objects. You may play with this example using the StateWORKS simulator (SWLab). This simulation tool which supports the development of a control system can be downloaded from this site. Using StateWORKS Studio you may change the behavior of the MWOven control and test it with SWLab. SWLab simulates inputs and outputs and contains the VFSM executor. To see what is going on, do not merely open the state machine and its states, but investigate the "Dictionary" and "Name" pull-down menus from the top tool-bar, as well as the open "Project" window to see how all the objects are defined and configured.

Demo

When you install the StateWORKS Studio you will find the entire project in the folder ..\Project\Examples-Web\ MWOven. You may inspect and change the project using the StateWORKS Studio. You may run and test the MWOven application using SWlab and SWMON or SWTerm.

Test: start SWLab and open the MWOven.swd file. SWLab displays then the DI inputs: Door and Run, DO outputs: Power and Lamp, and the NI input: CookingTime. As the CookingTime is initialized to 2048 you would like probably to set it to some lower value or 0 in the beginning. (Take no account of the scale markings, which are only intended as a logo for a numeric setting. You can quickly alter the setting by moving the pointer with a mouse select/move operation.) Alternatively, you may set the Offset property of the MW:Ni:CookinTime when specifying the RTDB to -2048.