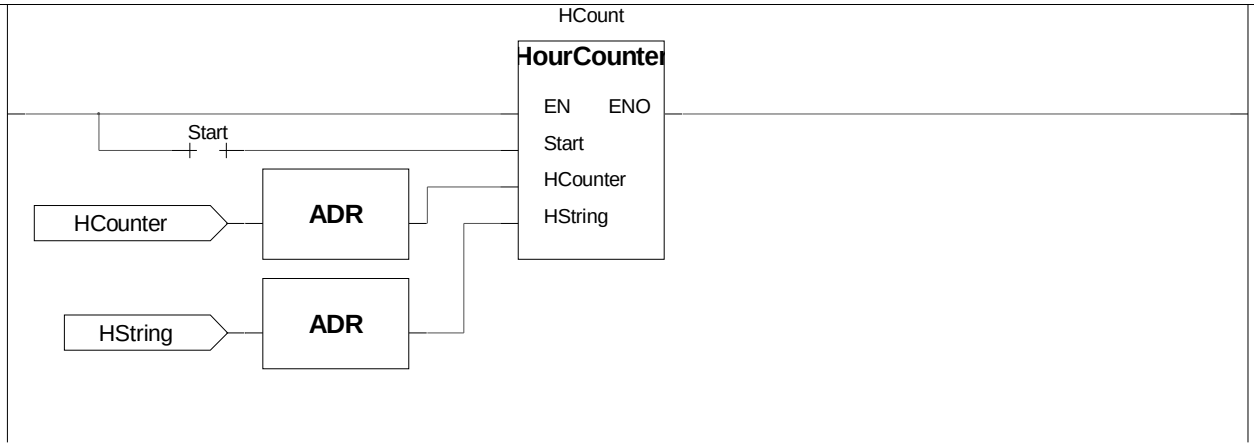
```
VAR_GLOBAL
Start AT %IX0.0 : BOOL; (* Start *)
HString : STRING[ 32 ]; (* Hour counter string report *)
END_VAR


VAR_GLOBAL RETAIN
HCounter : UDINT; (* Hour counter *)
END_VAR
```

PROGRAM Alma

```
VAR
HCount : HourCounter; (* Hours counter [h:m:s] *)
END_VAR
```

1



	Project : HoursCounter	
	PROGRAM : Alma	
	Release : Relevage1	Ver :1.00
	Author :	Date:27/03/2012
	Note :	Page:1 of 1

FUNCTION_BLOCK HourCounter

```
VAR_INPUT
Start : BOOL; (* Input  START  *)
HCounter : @UDINT; (* Hours counter address *)
HString : @USINT; (* Hours counter string address *)
END_VAR

VAR_EXTERNAL
SysClock1000 : BOOL; (* System clock 1000 mS *)
END_VAR

VAR
Pulse : BOOL; (* Time base pulse *)
i : INT; (* Aux counter *)
SecTrigger : BOOL; (* Second trigger *)
Hours : UDINT; (* Working hours *)
Minutes : USINT; (* Working minutes *)
Seconds : USINT; (* Working seconds *)
END_VAR
```

```
 1 (* ********************************************************************** *)
 2 (* FUNCTION BLOCK "HourCounter"                                           *)
 3 (* ********************************************************************** *)
 4 (* This FB calculates the work time of a device, the device start signal      *)
 5 (* must be passed.                                                        *)
 6 (*                                                                        *)
 7 (* Since the hours counter has to be RETAIN and is not possible to define     *)
 8 (* RETAIN variables in a FB, so the hours counter variable must be a RETAIN   *)
 9 (* variable the address of which must be passed to the FB.                *)
10 (*                                                                        *)
11 (* The FB returns also a string that reports the working time expressed as:   *)
12 (* "hhhhhhhhhh:mm:ss;"                                                    *)
13 (*                                                                        *)
14 (* Input variables:                                                       *)
15 (* Start: Device start signal.                                            *)
16 (* HCounter: Hours counter address.                                       *)
17 (* HString: Hours counter string address.                                 *)
18 (* ---------------------------------------------------------------------- *)
19     (* One shot varaiables management. *)
20
21     SecTrigger:=FALSE; (* Second trigger *)
22
23     (* Checks if a seconds is passed. *)
24
25     IF (SysClock1000 <> Pulse) THEN
26         Pulse:=SysClock1000; (* Time base pulse *)
27         SecTrigger:=TRUE; (* Second trigger *)
28     END_IF;
29
30     (* Checks if the device is working or not. *)
31
32     IF (Start) AND (SecTrigger) THEN
33         @HCounter:=@HCounter+1; (* Hours counter *)
34     END_IF;
35
36     (* Create the output string, every 1 second to save execution time. *)
37
38     IF NOT (SecTrigger) THEN RETURN; END_IF;
```

	Project : HoursCounter	
	FUNCTION BLOCK : HourCounter	
	Release : Relevage1	Ver :1.00
	Author :	Date:27/03/2012
	Note :	

```
39
40      (* Calculate time in hours, minutes and seconds. *)
41
42      Hours:=@HCounter/3600; (* Working hours *)
43      Minutes:=TO_USINT((@HCounter-(Hours*3600))/60); (* Working minutes *)
44      Seconds:=TO_USINT(@HCounter-(Hours*3600)-(Minutes*60)); (* Working seconds *)
45
46      (* Returns the time in a string. *)
47
48      i:=SysVarsnprintf(HString, 12, '%10d:',UDINT_TYPE, ADR(Hours));
49      i:=SysVarsnprintf(HString+11, 4, '%02d:',USINT_TYPE, ADR(Minutes));
50      i:=SysVarsnprintf(HString+14, 4, '%02d;',USINT_TYPE, ADR(Seconds));
51
52  (* [End of file] *)
53
54
```