Safety-Critical Systems 4:
Engineering of Embedded Software Systems
WiSe 2002/03

Jan Bredereke

due: Nov. 14, 2002

Solution for Assignment 3

Bicycle Odometer – Software Requirements Specification

This software requirements specification implements the example system requirements specification by Jan Bredereke for Assignment 2.

Recommendation: if you would like to understand how I designed this specification, compare it to the system requirements specification and find out which parts of the system requirements specification appear in which places here. One major task of design is to assign the parts of the system requirements specification to the components of the system, i.e., IN, SOF, and OUT in our case.

Environmental Quantities

This specification takes the environmental quantities specification by Jan Bredereke for Assignment 1 as a base, which is not repeated here. But we make one update to this specification: the value set of the controlled quantity ^{c}number is extended by the value ^{C}invalid. This value is an abstraction for all kinds of display that are not valid numbers.

Input and Output Quantities

In our particular application, there is no input or output variable that carries a unit. We therefore omit the corresponding column in the following two tables.

Input Quantities

Variable	Description	Value Set	Notes
^{i}inByte	bit field indicating rotation sensor and button position	$\{0b00000000, \ldots 0b11111111\}$	1

Notes

1. The variable ^{i}inByte has 8 bits with the following meaning: the least significant bit (at position 0) reflects the rotation sensor. It is 1 if the resistance was $< 10\ \Omega$ at measurement time, and 0 otherwise. The next bit (at position 1) reflects the multi-function button position. It is 1 if the button was down at measurement time, and 0 otherwise. The other six bits are always 0.

Output Quantities

Variable	Description	Value Set	Notes
^{o}numberOut	number displayed	$\times_{i=0}^{4}\{0,\ldots 255\}$	1
^{o}modeOut	mode indicator displayed	$\{0b00000000,\ldots 0b11111111\}$	2

Notes

1. The variable ^{o}numberOut[] is an array of five 8-bit variables. The array ranges from index 0 to index 4. Each of the variables determines the visible value of one digit on the odometer's display. ^{o}numberOut[0] is the most significant digit and ^{o}numberOut[4] is the least significant digit. Indices between these behave accordingly. A value of 0 displays the digit "0", a value of 9 displays the digit "9". Values between 0 and 9 behave accordingly. A value of 15 leaves the digit blank. The behaviour for other values is not defined.

2. The variable ^{o}modeOut has 8 bits with the following meaning: the bits 0, 1, and 2 reflect the indicators for "km/h", "km total", and "km trip", respectively. If a bit is 1, the corresponding indicator is visible, if the bit is 0, the indicator is not visible. When at least one of the bits 1 and 2 is set, the decimal point of the number display is visible, otherwise it is invisible. The remaining 5 bits must always be 0.

Input/Output Quantities

We assume that the basic operating system does not allow any concurrent execution. Therefore, at most one user routine will be running at any point of time.

Variable	Description	Value Set	Notes
^{io}rtmainRunning	user routine `rtmain()` is currently running	$\mathbb{B}$	
^{io}rtinitRunning	user routine `rtinit()` is currently running	$\mathbb{B}$	

Issues Related to the Entire System

Accuracy

The accuracy of long time intervals ($\geq$ 1 s), i.e., the clock drift, shall be $\pm 10^{-3}$ of the true value. This shall hold for any sub-interval of time.

In the following, all expressions referring to time must be interpreted not with respect to the true time t, but with respect to a drifted time reference $t_d(t)$.

Mode Class

^{Cl}control : ^{Md}osInBoot, ^{Md}rtinitInControl, ^{Md}osAfterBoot, ^{Md}rtmainInControl, ^{Md}osInControl

initial mode: ^{Md}osInBoot

Mode	Event Class	New Mode
^{Md}osInBoot	@T($t = {}^{C}$bootDrtn)	^{Md}rtinitInControl
^{Md}rtinitInControl	@F(^{io}rtinitRunning)	^{Md}osAfterBoot
^{Md}osAfterBoot	@T(t = First(@F(^{io}rtinitRunning)) + ^{C}afterBootDrtn)	^{Md}rtmainInControl
^{Md}rtmainInControl	@F(^{io}rtmainRunning)	^{Md}osInControl
^{Md}osInControl	t − Last(@T(^{io}rtmainRunning)) = 1 ms	^{Md}rtmainInControl

Event Class

name	event class
^{e}tick	@T(inmode(^{Md}rtmainInControl))

Auxiliary Functions

The relation from an speed output value to the speed display controlled value:

$^{f}\text{speedOut}(ov) = ov[2] \cdot 100 + ov[3] \cdot 10 + ov[4]$

Note: no decimal point is displayed, and the number is displayed flush right. If the speed is zero, a single 0 is displayed. Leading zeroes are not displayed.

The relation from a distance output value to a distance display controlled value:

$^{f}\text{distOut}(ov) = ov[0] \cdot 1000 + ov[1] \cdot 100 + ov[2] \cdot 10 + ov[3] + ov[4] \cdot 0.1$

Note: a decimal point is always displayed. The two digits immediately left and right of the decimal point are always displayed, even if they are 0. Otherwise, leading or trailing zeroes are not displayed. The number is displayed flush right.

Editing note: the type of the controlled variable ^{c}number should really be adapted to match the type of the output variable ^{o}numberOut better.

The Input Device Requirements Specification IN

Conditions

name	condition
^{p}sensorClosed	$^{m}\text{sensor} < 10\ \Omega$

Event Classes

name	event class
^{e}rtFuncLeave	@F(inmode(^{Md}rtmainInControl)) ∨ @F(inmode(^{Md}rtinitInControl))

Accuracy

The accuracy of the measurement of electrical resistance for ^{m}sensor shall be $\pm 3\ \Omega$.

Auxiliary Functions

Note: in the following, there is non-determinism due to the uncertainty about the point in time when polling for the input values takes place.

The set of possible polled values for the rotation sensor:

${}^{f}\text{sensorClosedPolled}(t) =$

$\text{p}_\text{T} : H_1$, $\text{r}_\text{T} : G$, Normal	
^{e}tick	$\{sc \mid \exists t_p \,.\, sc = {}^{p}\text{sensorClosed}(t_p) \wedge \text{Last}({}^{e}\text{rtFuncLeave}) < t_p < t\}$
$\neg {}^{e}$tick	$*$

The set of possible polled values for the button position:

${}^{f}\text{buttonPolled}(t) =$

$\text{p}_\text{T} : H_1$, $\text{r}_\text{T} : G$, Normal	
^{e}tick	$\{sc \mid \exists t_p \,.\, sc = ({}^{m}\text{button}(t_p) = {}^{C}\text{down}) \wedge \text{Last}({}^{e}\text{rtFuncLeave}) < t_p < t\}$
$\neg {}^{e}$tick	$*$

The set of possible values for the polled input byte:

${}^{f}\text{inBytePolled}(t) = \{n \mid n = 0b1 \cdot n_s \ + \ 0b10 \cdot n_b \wedge$
$n_s \in \text{BoolSetToNums}({}^{f}\text{sensorClosedPolled}(t)) \ \wedge \ n_b \in \text{BoolSetToNums}({}^{f}\text{buttonPolled}(t))\}$

Input Variables

${}^{i}\text{inByte}(t) \in$

$\text{p}_\text{T} : H_1$, $\text{r}_\text{T} : G$, Normal	
^{e}tick	${}^{f}\text{inBytePolled}(t)$
t(inmode(^{Md}rtmainInControl))	{ ^{i}inByte (Last(@T(inmode(^{Md}rtmainInControl)))) }
F(inmode(^{Md}rtmainInControl))	$*$

Input/Output Variables

@T(inmode(^{Md}rtinitInControl)) $\implies$ @T(^{io}rtinitRunning)

@T(inmode(^{Md}rtmainInControl)) $\implies$ @T(^{io}rtmainRunning)

The Output Device Requirements Specification OUT

Controlled Variables

^{c}modeInd =

$p_T : H_1$, $r_T : G$, Normal	
^{o}modeOut & 0b1	"km/h"
^{o}modeOut & 0b100	"km trip"
^{o}modeOut & 0b10	"km total"

^{c}number =

$p_T : H_1$, $r_T : G$, Normal	
inmode(^{Md}speed)	^{f}speedOut (^{o}NumberOut)
inmode(^{Md}dayTrip)	^{f}distOut (^{o}NumberOut)
inmode(^{Md}total)	^{f}distOut (^{o}NumberOut)

Tolerance

The update delay for ^{c}modeInd and ^{c}number must be less than 0.099 s, including the delay by the optical LCD component.

The Software Requirements Specification SOF

Event Classes

name	**event class**
^{e}roundComplete	^{e}tick WHEN($\forall e \in$ Prev10th(^{e}tick) . (^{i}inByte($e.t$)&0b1) = 1) WHEN(#(^{e}tick) $\geq$ 11) WHEN($\forall e \in$ Prev9(^{e}tick) . (^{i}inByte($e.t$)&0b1) = 0)
^{e}buttonPressed	^{e}tick WHEN($\forall e \in$ Prev9(^{e}tick) . (^{i}inByte($e.t$)&0b10) = 1)
^{e}buttonPressedLong	^{e}tick WHEN($\forall e \in$ Prev1999(^{e}tick) . (^{i}inByte($e.t$)&0b10) = 1)

Mode Class

^{Cl}btnDrtn : ^{Md}btnUp, ^{Md}btnShort, ^{Md}btnLong

initial mode: ^{Md}btnUp

Mode	**Event Class**	**New Mode**
^{Md}btnUp	^{e}buttonPressed	^{Md}btnShort
^{Md}btnShort	^{e}tick $\wedge$ $\neg$^{e}buttonPressed	^{Md}btnUp
	^{e}tick $\wedge$ ^{e}buttonPressedLong	^{Md}btnLong
^{Md}btnLong	^{e}tick $\wedge$ $\neg$^{e}buttonPressed	^{Md}btnUp

Event Classes

name	event class
^{e}switchMode	@F(^{Md}btnShort) $\wedge$ @T(^{Md}btnUp)
^{e}reset	@T(^{Md}btnLong)

Mode Class

^{Cl}display : ^{Md}speed, ^{Md}total, ^{Md}dayTrip

initial mode: ^{Md}speed

Mode	Event Class	New Mode
^{Md}speed	^{e}switchMode $\vee$ ^{e}reset	^{Md}dayTrip
^{Md}dayTrip	^{e}switchMode	^{Md}total
^{Md}total	^{e}switchMode	^{Md}speed
	^{e}reset	^{Md}dayTrip

maximum delay: any change happening at an event ^{e}tick shall become visible before the event @F(inmode(^{Md}rtmainInControl)).

Auxiliary Functions

${}^{f}\text{totalPulses} = \#({}^{e}\text{roundComplete})$

${}^{f}\text{totalDist} = \text{Round1}({}^{f}\text{totalPulses} \cdot {}^{C}\text{circumference}) \mod {}^{C}\text{numberLimit}$

${}^{f}\text{dayTripPulses} = \#({}^{e}\text{roundComplete}) - \#(\text{Prev}({}^{e}\text{roundComplete}, \text{Last}({}^{e}\text{reset})))$

${}^{f}\text{dayTripDist} = \text{Round1}({}^{f}\text{dayTripPulses} \cdot {}^{C}\text{circumference}) \mod {}^{C}\text{numberLimit}$

^{f}pulsePeriod =

$p_T : H_1,\ r_T : G$, Normal	
$\#({}^{e}\text{roundComplete}) \geq 2$	$\text{Last}({}^{e}\text{roundComplete}) - \text{SecondButLast}({}^{e}\text{roundComplete})$
$\#({}^{e}\text{roundComplete}) < 2$	^{C}veryLongPulse

The resolution of time measurement for ^{f}pulsePeriod shall be 1 ms.

${}^{f}\text{speedRaw} = \text{Round}(3.6 \frac{\text{km/h}}{\text{m/s}} \ \frac{{}^{C}\text{circumference}}{{}^{f}\text{pulsePeriod}})$

^{f}speedDisp =

$p_T : H_1,\ r_T : G$, Normal	
${}^{f}\text{pulsePeriod} < 3.6 \frac{\text{km/h}}{\text{m/s}} \ \frac{{}^{C}\text{circumference}}{1\ \text{km/h}}$	^{f}speedRaw
${}^{f}\text{pulsePeriod} \geq 3.6 \frac{\text{km/h}}{\text{m/s}} \ \frac{{}^{C}\text{circumference}}{1\ \text{km/h}}$	0 km/h

Note: the above cut-off at 1 km/h avoids the problem of a display not always returning to zero during stand-still of the bicycle, which is annoying.

Output Variables

^{o}modeOut =

$p_T : H_1$, $r_T : G$, Normal	
inmode(^{Md}speed)	$0b00000001$
inmode(^{Md}dayTrip)	$0b00000100$
inmode(^{Md}total)	$0b00000010$

^{o}numberOut =

$p_T : H_1$, $r_T : G$, Normal	
inmode(^{Md}speed)	${}^{f}\text{speedOut}^{-1}({}^{f}\text{speedDisp})$
inmode(^{Md}dayTrip)	${}^{f}\text{speedOut}^{-1}({}^{f}\text{dayTripDist})$
inmode(^{Md}total)	${}^{f}\text{speedOut}^{-1}({}^{f}\text{totalDist})$

Input/Output Variables

Processing of user routines is "fast":

$\exists t_r \,.\, 0\text{ ms} < t_r \ll 1\text{ ms} \wedge @\text{T}({}^{io}\text{rtinitRunning}(t)) \implies @\text{F}({}^{io}\text{rtinitRunning}(t + t_r))$

$\exists t_r \,.\, 0\text{ ms} < t_r \ll 1\text{ ms} \wedge @\text{T}({}^{io}\text{rtmainRunning}(t)) \implies @\text{F}({}^{io}\text{rtmainRunning}(t + t_r))$

Dictionary

Constants

name	value	description
^{C}bootDrtn	$\in [0\text{ s} \ldots 1\text{ s}]$	duration of OS boot.
^{C}afterBootDrtn	$\in [0\text{ s} \ldots 1\text{ ms}]$	delay between end of `rtinit()` and first call to `rtmain()`.
^{C}circumference	0.711 m	Circumference of the wheel.
^{C}numberLimit	10000.0	Smallest number which is too large to display.
^{C}veryLongPulse	1000 s	This inter-pulse time means "very long" and will result in a speed display of 0 km/h.

Mathematic Functions

$\text{SecondButLast}(e) = \text{Last}(e, \text{Last}(e))$

$\text{Round} : \mathbb{R}_0^+ \mapsto \mathbb{N}, \qquad \forall x \in \mathbb{R} \,.\, -0.5 \leq x - \text{Round}(x) < 0.5$

$\text{Round1} : \mathbb{R}_0^+ \mapsto \mathbb{R}_0^+, \qquad \forall x \in \mathbb{R} \,.\, \text{Round1}(x) = \text{Round}(10\ x)/10$

$\text{BoolSetToNums} : \mathbb{P}(\{\text{false}, \text{true}\}) \mapsto \mathbb{P}(\{0, 1\})$
$\text{BoolSetToNums} = \{(bs, is) \mid (\text{false} \in bs \Leftrightarrow 0 \in is) \wedge (\text{true} \in bs \Leftrightarrow 1 \in is)\}$

Prev9 : Event Classes, $\mathbb{R} \mapsto$ Event Classes
Prev9(e, t) delivers the set of the nine events of event class e that occur prior to time t. If there are less such events, the resulting set is accordingly smaller.
As usual, Prev9$(e) =$ Prev9(e, t_f)

Prev1999 : Event Classes, $\mathbb{R} \mapsto$ Event Classes
Prev1999(e, t) delivers the set of the 1999 events of event class e that occur prior to time t. If there are less such events, the resulting set is accordingly smaller.
As usual, Prev1999(e) = Prev1999(e, t_f)

Prev10th : Event Classes, $\mathbb{R} \mapsto$ Event Classes
Prev10th(e, t) delivers the set containing the tenth last event of event class e that occurs prior to time t. If there are less such events, the resulting set is accordingly empty.
As usual, Prev10th(e) = Prev10th(e, t_f)

The other mathematic functions are standard and are interpreted as in the lecture.