Testspezifikation-Fahrprogramm

Für das studentische Projekt *Sichere Eisenbahnsteuerung*

Datum	28.04.2010
Quelle	Dokumente → 04_Test → 04.01_Testspezifikation
Autoren	Norman Nieß
	Kai Dziembala
Version	0.1
Status	zum Review freigegeben

1 Historie

Version	Datum	Autor	Bemerkung
0.0	21.04.2010	Kai Dziembala Norman Nieß	Initialisierung der Testspezifikation
0.1	28.04.2010	Kai Dziembala Norman Nieß	Erstellung der Kapitel 3.3 – 5.5; Überarbeitung der Kapitel 3.1 und 3.2

2 Inhaltsverzeichnis

3 Testfall 1 „Fahrprogramm für Lokomotive 1“

3.1 Identifikation des Testobjektes

Es wird der Programmcode zum Softwaremodul „Fahrprogramm“ Version 0.2 getestet:

- Fahrprogramm.c
- Fahrprogramm.h

Das zu testende Modul enthält alle relevanten Fahranweisungen einer Fahraufgabe. Realisiert werden diese anhand eines Ringpuffers, dessen feste Abfolge von Einträgen bzw. Objekten eine Fahraufgabe abbilden. Grundlage für die Größe des Puffers ist ein detailliertes Gleislayout bzw. Streckenabbild und die Komplexität der Fahraufgabe. Der Ringpuffer wird sequentiell abgearbeitet.

3.2 Test-Identifikation

Testname:	Test_Fahrprogramm_Lok1
Szenario:	Das Modul Fahrprogramm wird wiederholt mit der Codierung für die Lokomotive 1 aufgerufen, wodurch nacheinander alle Fahranweisungen für diese Lokomotive zurückgegeben werden.
Verzeichnisse	
Testskripts:	Google Code → 04_Tests → 04.02_Testskript → Fahrprogramm
Testprotokolle:	Google Code → 04_Tests → 04.03_Testprotokolle → Fahrprogramm

3.3 Testskript

Es wird getestet, ob der Aufruf der Funktion 'getCommand(Byte lok)' mit den Übergabeparametern für die Lokomotive 1 ('0x0') die korrekte Fahranweisung zurückgibt.

Zunächst wird das Modul 'Fahrprogramm' initialisiert. Im Anschluss erfolgt in einer for-Schleife mit 10 Durchläufen die Abfrage der Fahranweisungen. Nach dieser Abfrage werden dessen Rückgabewerten mit den Erwarteten verglichen und das Vergleichsergebnis in der Konsole ausgegeben.

Dies wird mit folgendem Test-Skript realisiert:

3.4 Testreferenz

Während des Testdurchlaufs werden die in Tabelle 1 dargestellten Rückgabewerte erwartet:

Schleifendurchlauf	Fahranweisung	Byte 1	Byte 2
1	1	0000 0000	0000 0100
2	2	0000 0000	0000 0101
3	3	0000 0000	0000 0110
4	4	0000 0000	0000 0001
5	5	0000 0000	0000 0111
6	1	0000 0000	0000 0100
7	2	0000 0000	0000 0101
8	3	0000 0000	0000 0110
9	4	0000 0000	0000 0001
10	5	0000 0000	0000 0111

Tabelle 1: erwartete Rückgabewerte für den Testfall 1

3.5 Test-Protokoll

Das Konsolen-Ergebnis wird in das Dokument 'Protokoll_Test_Fahrprogramm' kopiert und diese Datei im Ordner 'Google Code → 04_Tests → 04.03_Testprotokolle → Fahrprogramm' abgelegt.

4 Testfall 2 „Fahrprogramm für Lokomotive 2“

4.1 Identifikation des Testobjektes

Es wird der Programmcode zum Softwaremodul „Fahrprogramm“ Version 0.2 getestet:

- Fahrprogramm.c
- Fahrprogramm.h

Das zu testende Modul enthält alle relevanten Fahranweisungen einer Fahraufgabe. Realisiert werden diese anhand eines Ringpuffers, dessen feste Abfolge von Einträgen bzw. Objekten eine Fahraufgabe abbilden. Grundlage für die Größe des Puffers ist ein detailliertes Gleislayout bzw. Streckenabbild und die Komplexität der Fahraufgabe. Der Ringpuffer wird sequentiell abgearbeitet.

4.2 Test-Identifikation

Testname:	Test_Fahrprogramm_Lok2
Szenario:	Das Modul Fahrprogramm wird wiederholt mit der Codierung für die Lokomotive 2 aufgerufen, wodurch nacheinander alle Fahranweisungen für diese Lokomotive zurückgegeben werden.
Verzeichnisse	
Testskripts:	Google Code → 04_Tests → 04.02_Testskript → Fahrprogramm
Testprotokolle:	Google Code → 04_Tests → 04.03_Testprotokolle → Fahrprogramm

4.3 Testskript

Es wird getestet, ob der Aufruf der Funktion 'getCommand(Byte lok)' mit den Übergabeparametern für die Lokomotive 2 ('0x1') die korrekte Fahranweisung zurückgibt.

Zunächst wird das Modul 'Fahrprogramm' initialisiert. Im Anschluss erfolgt in einer for-Schleife mit 56 Durchläufen die Abfrage der Fahranweisungen. Nach dieser Abfrage werden dessen Rückgabewerten mit den Erwarteten verglichen und das Vergleichsergebnis in der Konsole ausgegeben.

Dies wird mit folgendem Test-Skript realisiert:

4.4 Testreferenz

Während des Testdurchlaufs werden die in Tabelle 2 dargestellten Rückgabewerte erwartet:

Schleifendurchlauf	Fahranweisung	Byte 1	Byte 2
1	1	0000 0001	0000 0001
2	2	0000 0001	0000 0111
3	3	0000 0001	0000 0100
4	4	0000 0001	0000 0011
5	5	0000 0011	0000 0010
6	6	0000 0001	0000 0001
7	7	0000 0001	0000 1000
8	8	0000 0101	0000 1001
9	9	0000 0001	0000 1000
10	10	0000 0001	0000 0111
11	11	0000 0001	0000 0001
12	12	0000 0001	0000 0010
13	13	0001 0111	0000 0010
14	14	0000 0001	0000 0001
15	15	0000 0001	0000 1000
16	16	0000 0011	0000 1001
17	17	0000 0001	0000 1000
18	18	0000 0001	0000 0001
19	19	0000 0001	0000 0010
20	20	0000 0001	0000 0011
21	21	0000 0101	0000 0010
22	22	0000 0001	0000 0011
23	23	0000 0001	0000 0100

24	24	0000 0001	0000 0101
25	25	0000 0001	0000 0110
26	26	0000 0001	0000 0001
27	27	0000 0001	0000 1000
28	28	0001 0111	0000 1000
29	1	0000 0001	0000 0001
30	2	0000 0001	0000 0111
31	3	0000 0001	0000 0100
32	4	0000 0001	0000 0011
33	5	0000 0011	0000 0010
34	6	0000 0001	0000 0001
35	7	0000 0001	0000 1000
36	8	0000 0101	0000 1001
37	9	0000 0001	0000 1000
38	10	0000 0001	0000 0111
39	11	0000 0001	0000 0001
40	12	0000 0001	0000 0010
41	13	0001 0111	0000 0010
42	14	0000 0001	0000 0001
43	15	0000 0001	0000 1000
44	16	0000 0011	0000 1001
45	17	0000 0001	0000 1000
46	18	0000 0001	0000 0001
47	19	0000 0001	0000 0010

48	20	0000 0001	0000 0011
49	21	0000 0101	0000 0010
50	22	0000 0001	0000 0011
51	23	0000 0001	0000 0100
52	24	0000 0001	0000 0101
53	25	0000 0001	0000 0110
54	26	0000 0001	0000 0001
55	27	0000 0001	0000 1000
56	28	0001 0111	0000 1000

Tabelle 2: erwartete Rückgabewerte für den Testfall 2

4.5 Test-Protokoll

Das Konsolen-Ergebnis wird in das Dokument 'Protokoll_Test_Fahrprogramm' kopiert und diese Datei im Ordner 'Google Code → 04_Tests → 04.03_Testprotokolle → Fahrprogramm' abgelegt.

5 Testfall 3 „Fahrprogramm für eine nicht definierte Lokomotive“

5.1 Identifikation des Testobjektes

Es wird der Programmcode zum Softwaremodul „Fahrprogramm“ Version 0.2 getestet:

- Fahrprogramm.c
- Fahrprogramm.h

Das zu testende Modul enthält alle relevanten Fahranweisungen einer Fahraufgabe. Realisiert werden diese anhand eines Ringpuffers, dessen feste Abfolge von Einträgen bzw. Objekten eine Fahraufgabe abbilden. Grundlage für die Größe des Puffers ist ein detailliertes Gleislayout bzw. Streckenabbild und die Komplexität der Fahraufgabe. Der Ringpuffer wird sequentiell abgearbeitet.

5.2 Test-Identifikation

Testname:	Test_Fahrprogramm_nichtdefLok
Szenario:	Das Modul Fahrprogramm wird wiederholt mit einer nicht definierten Codierung aufgerufen, wodurch jeweils der Wert '0xFF' zurückgegeben werden muss.
Verzeichnisse	
Testskripts:	Google Code → 04_Tests → 04.02_Testskript → Fahrprogramm
Testprotokolle:	Google Code → 04_Tests → 04.03_Testprotokolle → Fahrprogramm

5.3 Testskript

Es wird getestet, ob der Aufruf der Funktion 'getCommand(Byte lok)' mit nicht definierten Parametern den Wert '0xFF' zurückgibt.

Zunächst wird das Modul 'Fahrprogramm' initialisiert. Im Anschluss erfolgt in einer for-Schleife mit 5 Durchläufen der Funktionsaufruf 'getCommand(Byte lok)' mit den im Kapitel 5.4 festgelegten Übergabeparametern. Nach dieser Abfrage sollte der Rückgabewert immer '0xFF' sein. Zum Schluss wird das Testergebnis in der Konsole ausgegeben.

Dies wird mit folgendem Test-Skript realisiert:

5.4 Testreferenz

Während des Testdurchlaufs werden die in Tabelle 3 dargestellten Rückgabewerte erwartet:

Schleifendurchlauf	Übergabeparameter	Rückgabewert
1	Byte '0x3'	'0xFF'
2	Byte '0x17'	'0xFF'
3	Byte '0x71'	'0xFF'
4	Byte '0xFF'	'0xFF'
5	String 'Test'	'0xFF'

Tabelle 3: erwartete Rückgabewerte für den Testfall 3

5.5 Test-Protokoll

Das Konsolen-Ergebnis wird in das Dokument 'Protokoll_Test_Fahrprogramm' kopiert und diese Datei im Ordner 'Google Code → 04_Tests → 04.03_Testprotokolle → Fahrprogramm' abgelegt.

6 Auswertung