FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

Analyse und Optimierung der internen Kommunikation eines Glucose-Puls-Reaktor-Programms geschrieben in LabVIEW

26.01.2018

Jessica Ahring

Inhaltsverzeichnis

LabVIEW

LabVIEW

> **Lab**oratory

LabVIEW

- **Lab**oratory
- **V**irtual

LabVIEW

- **Lab**oratory
- **V**irtual
- **I**nstrument

LabVIEW

- **Lab**oratory
- **V**irtual
- **I**nstrument
- **E**ngineering

LabVIEW

- **Lab**oratory
- **V**irtual
- **I**nstrument
- **E**ngineering
- **W**orkbench

Kurzübersicht über die Programmiersprache

Kurzübersicht über die Programmiersprache

> Entwicklungsumgebung

Kurzübersicht über die Programmiersprache

- Entwicklungsumgebung
- grafische Programmiersprache

Kurzübersicht über die Programmiersprache

- Entwicklungsumgebung
- grafische Programmiersprache
- Programmierung auf Blockdiagramm

Kurzübersicht über die Programmiersprache

- Entwicklungsumgebung
- grafische Programmiersprache
- Programmierung auf Blockdiagramm
- Anzeige auf Frontpanel

Kurzübersicht über die Programmiersprache

- Entwicklungsumgebung
- grafische Programmiersprache
- Programmierung auf Blockdiagramm
- Anzeige auf Frontpanel
- entwickelt von National Instruments (NI)

Kurzübersicht über die Programmiersprache

- Entwicklungsumgebung
- grafische Programmiersprache
- Programmierung auf Blockdiagramm
- Anzeige auf Frontpanel
- entwickelt von National Instruments (NI)
- VI = **v**irtuelle **I**nstrumente

Beispiel: Addition in LabVIEW

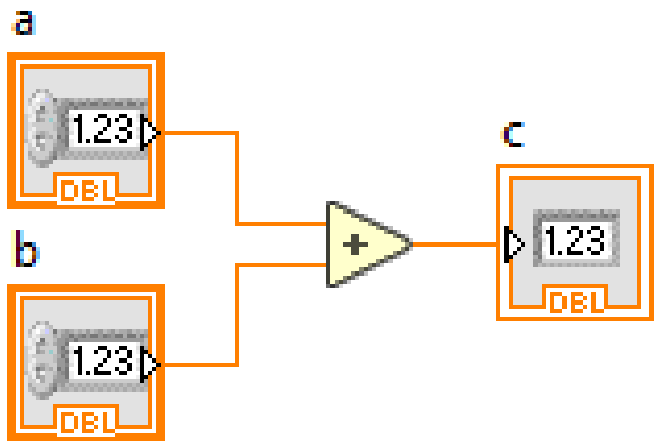


Abbildung: Blockdiagramm einer Addition in LabVIEW

Beispiel: Addition in LabVIEW

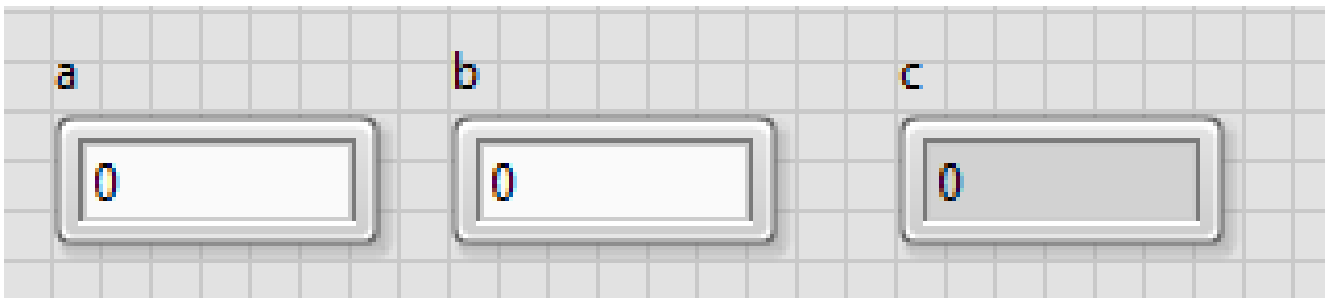


Abbildung: Frontpanel einer Addition in LabVIEW

Datenflussprinzip

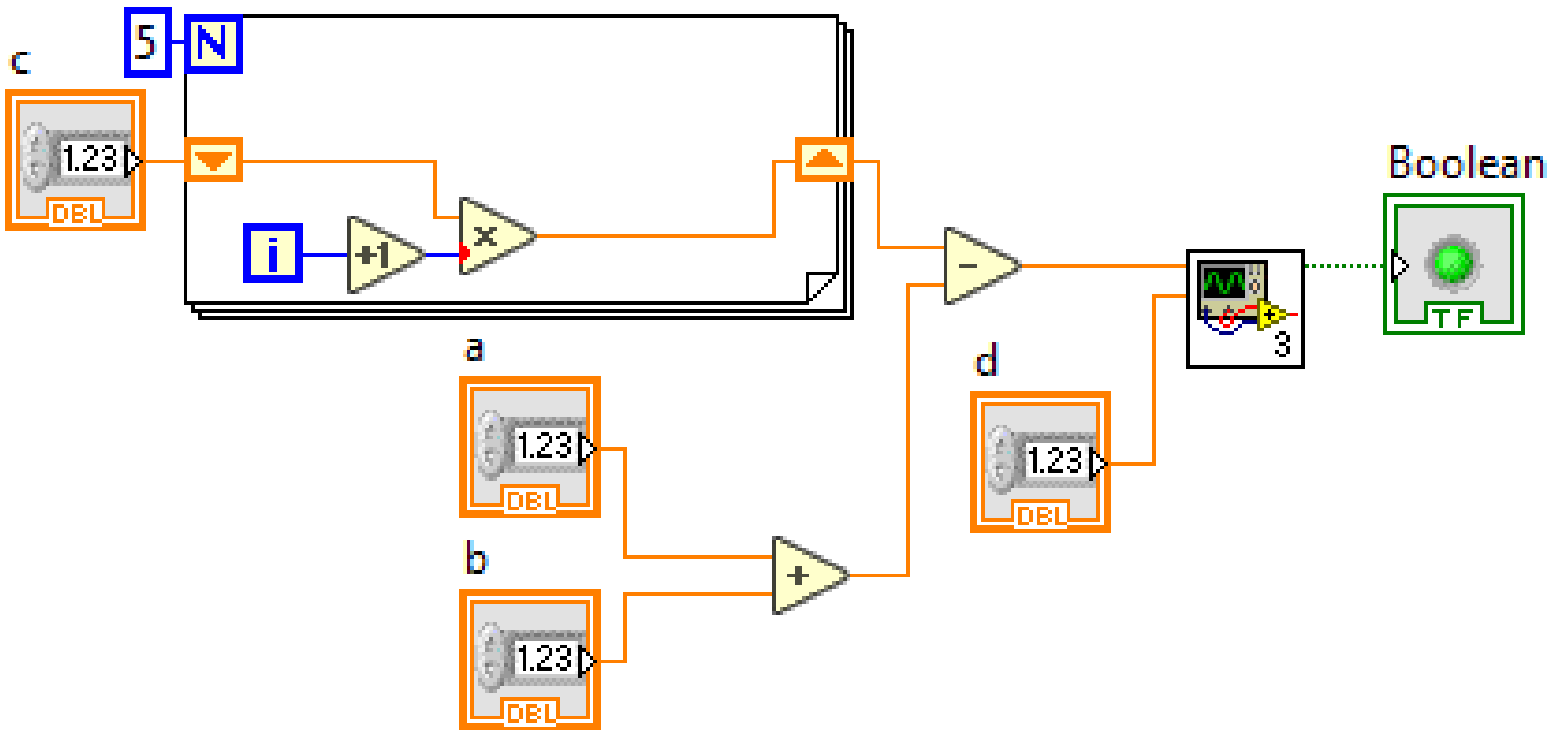


Abbildung: Datenflussprinzip

Wieso LabVIEW?

Wieso LabVIEW?

> Ursprünglich für Messrechner entwickelt

Wieso LabVIEW?

> Ursprünglich für Messrechner entwickelt

> Im IBG-1 Standardsprache für Messen, Steuern und Regeln

Wieso LabVIEW?

- Ursprünglich für Messrechner entwickelt
- Im IBG-1 Standardsprache für Messen, Steuern und Regeln
- Viele parallele Schleifen benötigt

Wieso LabVIEW?

- Ursprünglich für Messrechner entwickelt
- Im IBG-1 Standardsprache für Messen, Steuern und Regeln
- Viele parallele Schleifen benötigt
- Automatische Entstehung der Benutzeroberfläche

Wieso LabVIEW?

- Ursprünglich für Messrechner entwickelt
- Im IBG-1 Standardsprache für Messen, Steuern und Regeln
- Viele parallele Schleifen benötigt
- Automatische Entstehung der Benutzeroberfläche
- Flexibel einsetzbar

Wieso LabVIEW?

- Ursprünglich für Messrechner entwickelt
- Im IBG-1 Standardsprache für Messen, Steuern und Regeln
- Viele parallele Schleifen benötigt
- Automatische Entstehung der Benutzeroberfläche
- Flexibel einsetzbar
- Hardware von NI und Software arbeiten gut zusammen

Methoden des Datenaustauschs

Methoden des Datenaustauschs

> Variablen, Queues, etc. enthalten

Methoden des Datenaustauschs

> Variablen, Queues, etc. enthalten

> mehr Flexibilität

Methoden des Datenaustauschs

- Variablen, Queues, etc. enthalten
- mehr Flexibilität
- Durchbrechen Datenflussprinzip

Methoden des Datenaustauschs

- Variablen, Queues, etc. enthalten
- mehr Flexibilität
- Durchbrechen Datenflussprinzip
- Erhöhte Komplexität

Methoden des Datenaustauschs

- Variablen, Queues, etc. enthalten
- mehr Flexibilität
- Durchbrechen Datenflussprinzip
- Erhöhte Komplexität
- Erhöhung des Speicherbedarfs des Programms

Methoden des Datenaustauschs

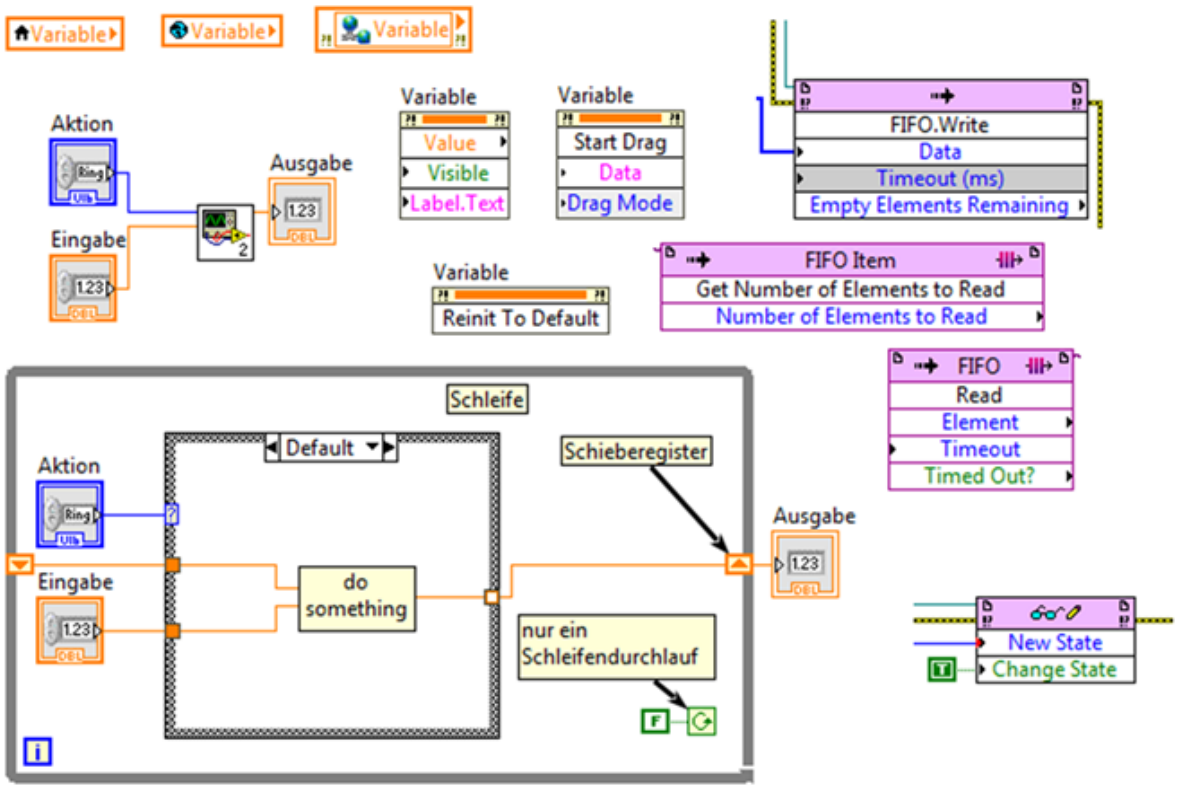


Abbildung: Datenübertragungstypen 1

Methoden des Datenaustauschs

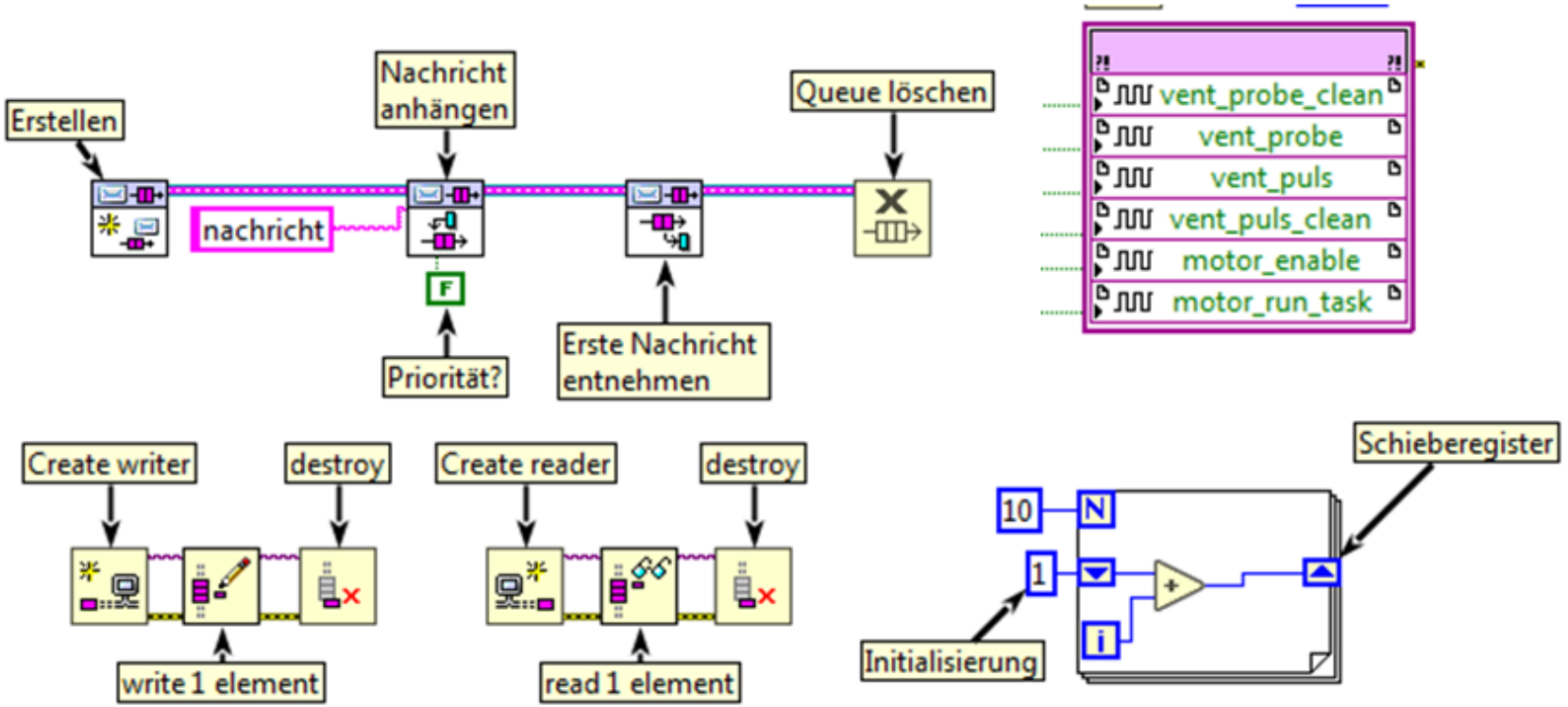


Abbildung: Datenübertragungstypen 2

Glucose-Puls-Reaktor

Glucose-Puls-Reaktor

> 1 Liter Behälter

Glucose-Puls-Reaktor

- 1 Liter Behälter
- Pumpen, Sensoren, Schläuche

Glucose-Puls-Reaktor

- 1 Liter Behälter
- Pumpen, Sensoren, Schläuche
- Schnelle Probennahme

Glucose-Puls-Reaktor

- 1 Liter Behälter
- Pumpen, Sensoren, Schläuche
- Schnelle Probennahme
- Erkenntnisse über Reaktionsverläufe

Glucose-Puls-Reaktor-Programm

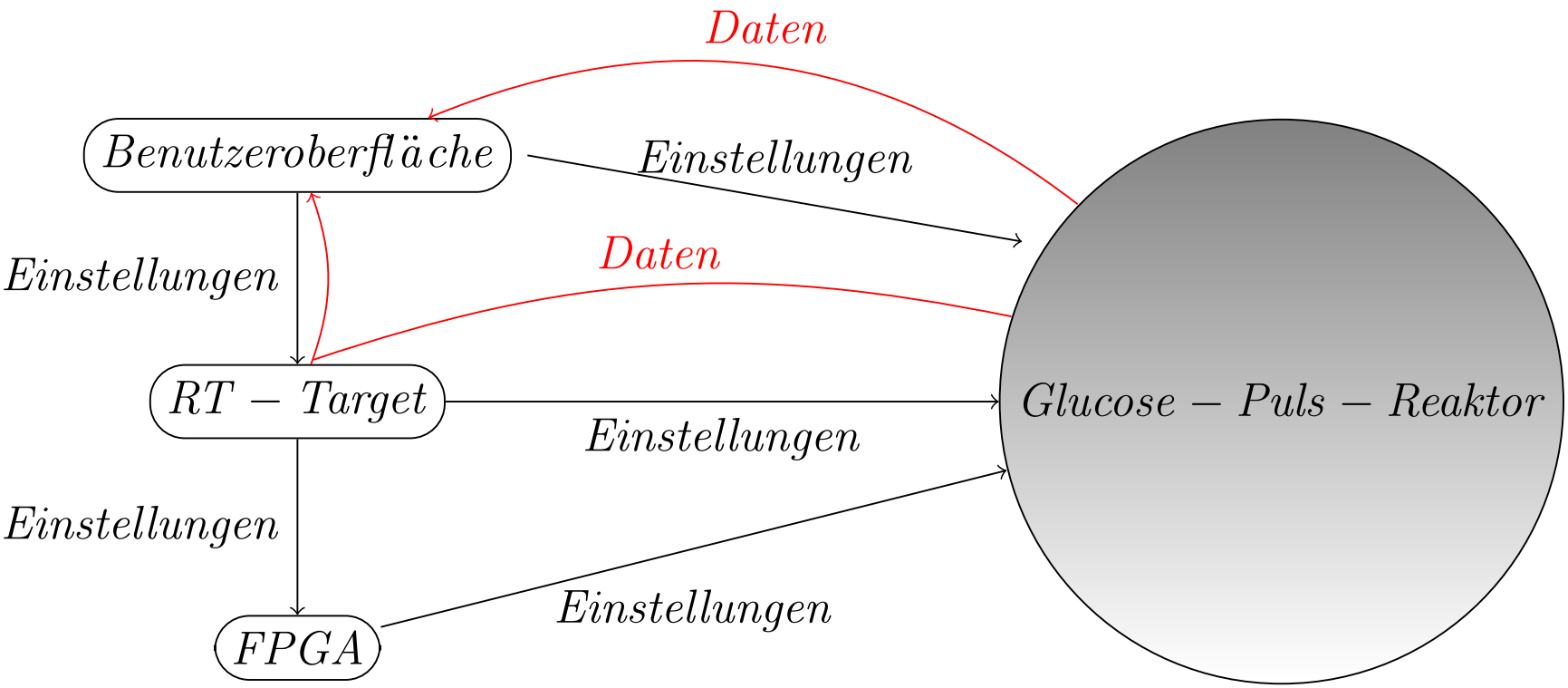


Abbildung: Kommunikation innerhalb des Programms und zum Reaktor

Motivation

> zuverlässige und schnelle Kommunikation benötigt

Motivation

> zuverlässige und schnelle Kommunikation benötigt

> Funktionsfähige Kommunikation vorhanden

Motivation

> zuverlässige und schnelle Kommunikation benötigt
> Funktionsfähige Kommunikation vorhanden
> Optimierungspotential:

Motivation

- zuverlässige und schnelle Kommunikation benötigt
- Funktionsfähige Kommunikation vorhanden
- Optimierungspotential:
  - Speicherplatz

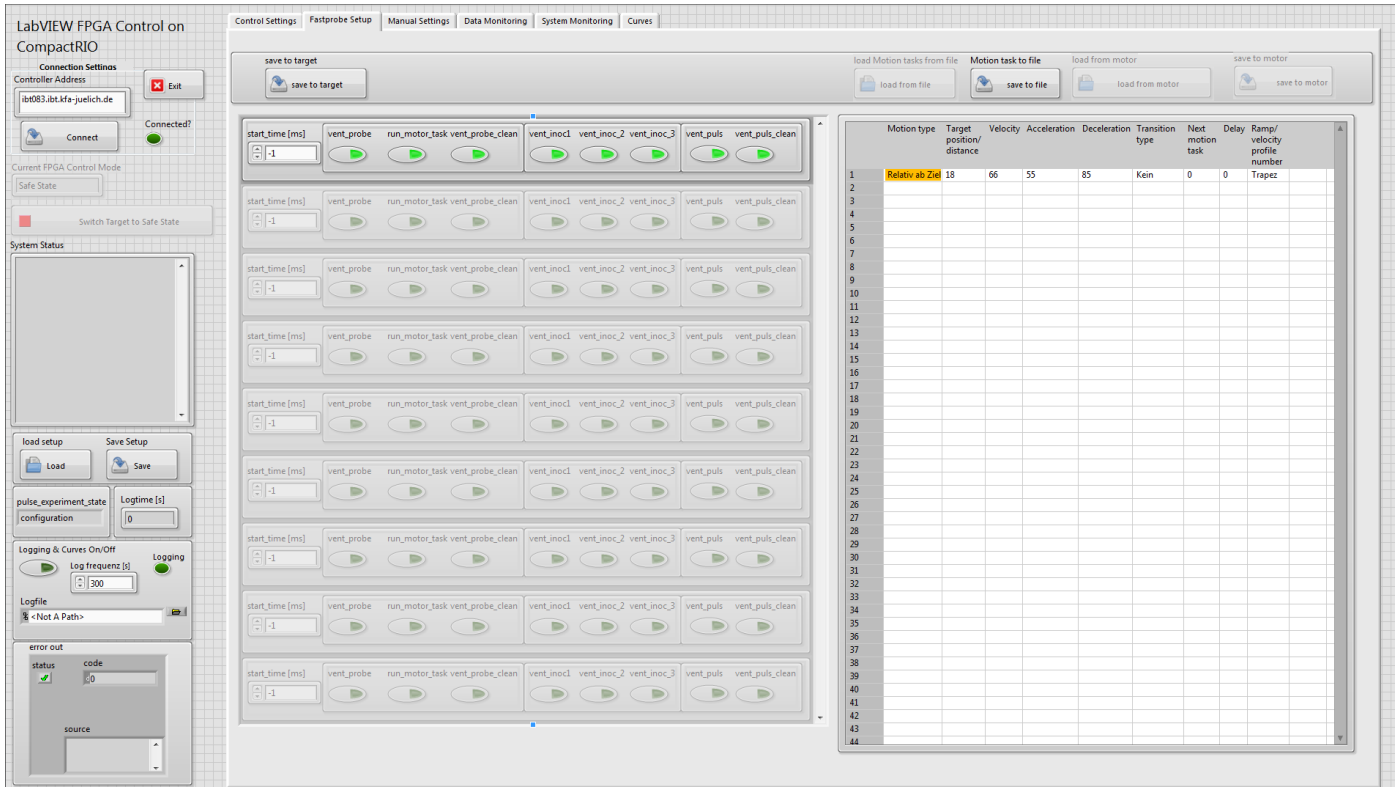
Motivation

- zuverlässige und schnelle Kommunikation benötigt
- Funktionsfähige Kommunikation vorhanden
- Optimierungspotential:
  - Speicherplatz
  - Redundanzen

Motivation

- zuverlässige und schnelle Kommunikation benötigt
- Funktionsfähige Kommunikation vorhanden
- Optimierungspotential:
  - Speicherplatz
  - Redundanzen
  - Zeit

Benutzeroberfläche



Benutzeroberfläche

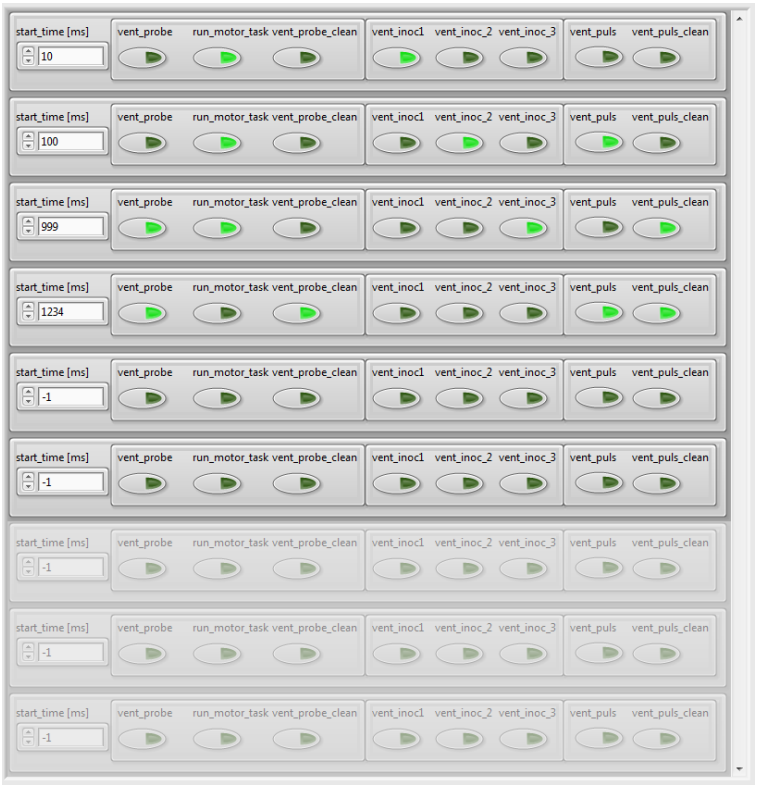


Abbildung: Feld zur Benutzereingabe der Ventilschaltungen

Benutzeroberfläche - RT-Target

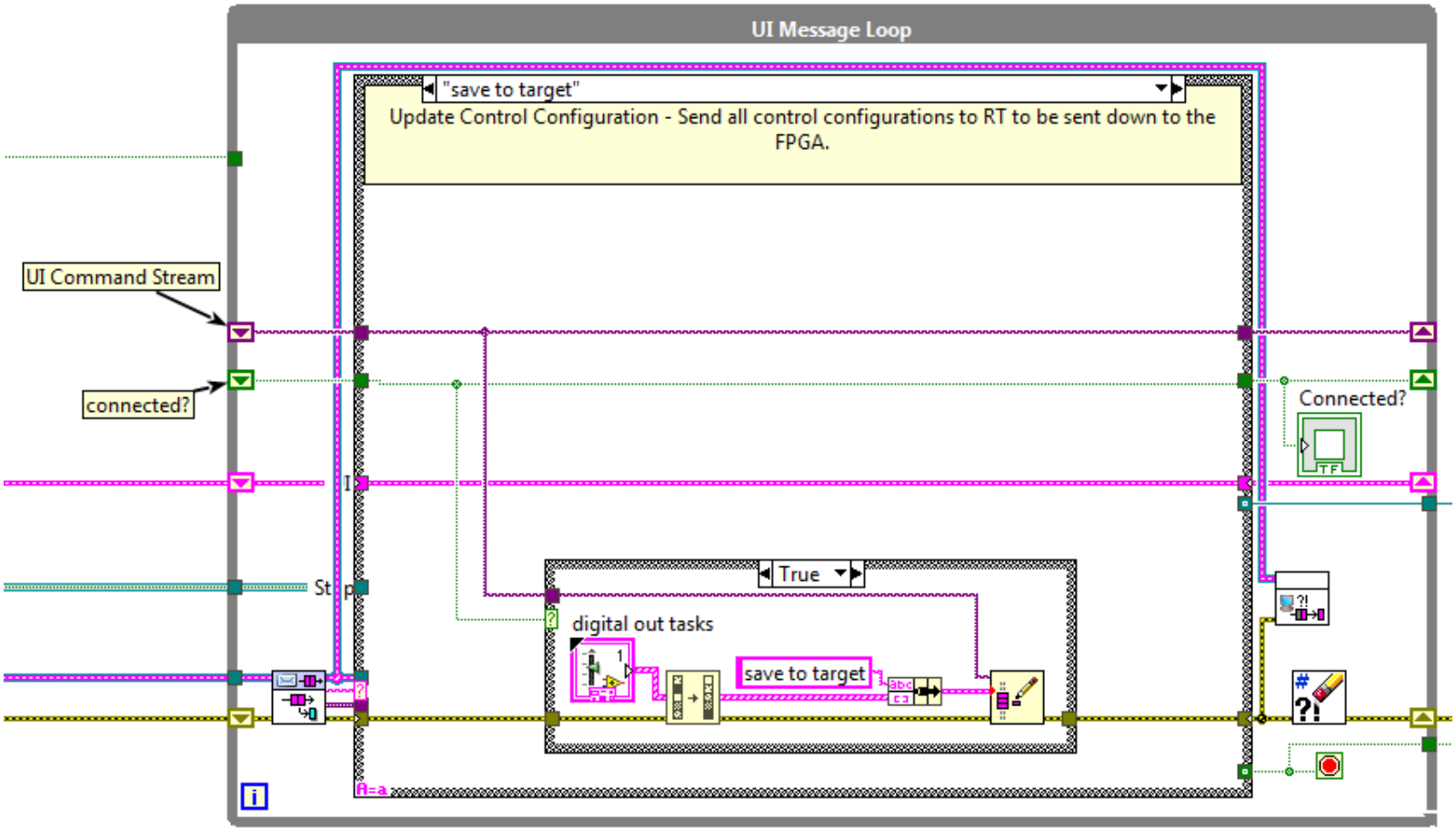


Abbildung: UI Main.vi

Benutzeroberfläche - RT-Target

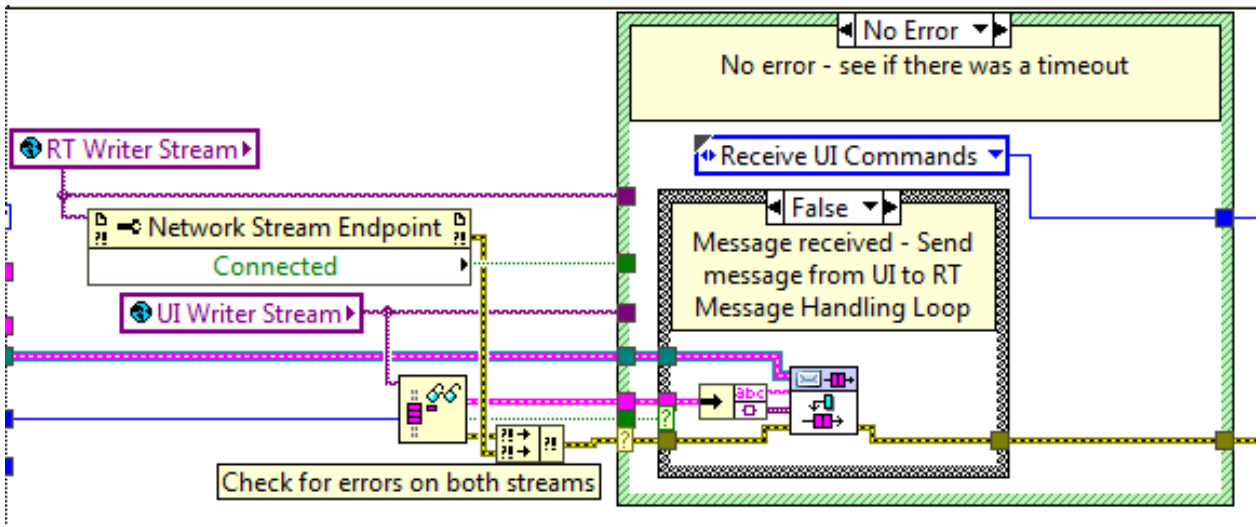


Abbildung: RT Loop-UI Commands.vi

RT-Target - Benutzeroberfläche

Verschiedene Kommunikationsmöglichkeiten:

RT-Target - Benutzeroberfläche

Verschiedene Kommunikationsmöglichkeiten:

1. Stream => Queue

RT-Target - Benutzeroberfläche

Verschiedene Kommunikationsmöglichkeiten:

1. Stream => Queue
2. 12 Umgebungsvariablen

RT-Target - Benutzeroberfläche

Verschiedene Kommunikationsmöglichkeiten:

1. Stream => Queue
2. 12 Umgebungsvariablen

Keine Optimierung möglich

RT-Target

RT-Target

> Steuerung des RT-Targets: „RT Main.vi“

RT-Target

> Steuerung des RT-Targets: „RT Main.vi“
> Verwendung von drei SubVIs

RT-Target

- Steuerung des RT-Targets: „RT Main.vi“
- Verwendung von drei SubVIs
- Verwendung einer gemeinsamen Queue

RT-Target

- Steuerung des RT-Targets: „RT Main.vi“
- Verwendung von drei SubVIs
- Verwendung einer gemeinsamen Queue
- 22 Zugriffe auf globale Variablen

RT-Target

- Steuerung des RT-Targets: „RT Main.vi“
- Verwendung von drei SubVIs
- Verwendung einer gemeinsamen Queue
- 22 Zugriffe auf globale Variablen
- 3 Eigenschaftsknoten

RT-Target

- Steuerung des RT-Targets: „RT Main.vi“
- Verwendung von drei SubVIs
- Verwendung einer gemeinsamen Queue
- 22 Zugriffe auf globale Variablen
- 3 Eigenschaftsknoten
- keine lokalen Variablen

RT-Target

- Steuerung des RT-Targets: „RT Main.vi“
- Verwendung von drei SubVIs
- Verwendung einer gemeinsamen Queue
- 22 Zugriffe auf globale Variablen
- 3 Eigenschaftsknoten
- keine lokalen Variablen
- **Keine Optimierung möglich**

RT-Target - FPGA

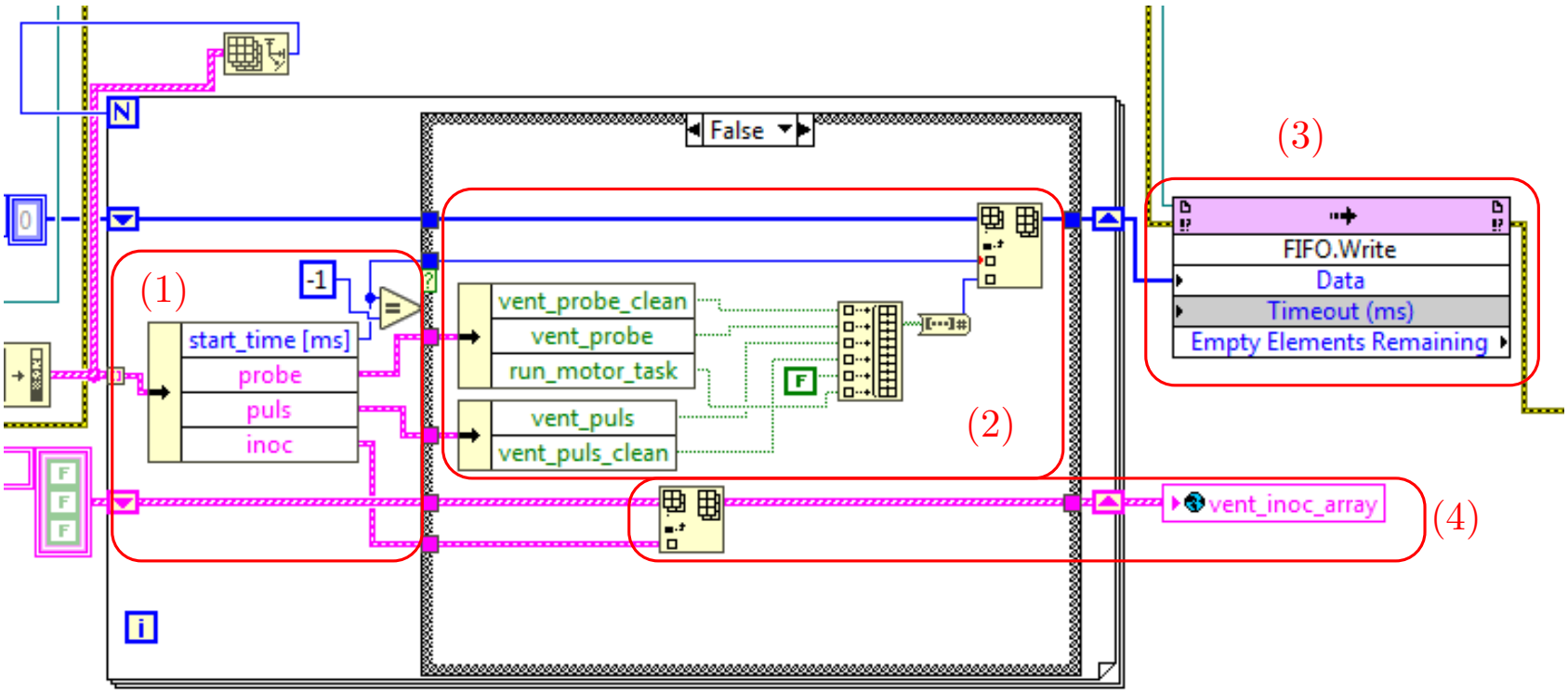


Abbildung: RT Main.vi

RT-Target - FPGA

Tabelle: Umwandlung eines Boolean-Arrays in eine Zahl – Beispiel

vent_probe_clean	vent_probe	vent_puls	vent_puls_clean	False	run_motor_task	Dualzahl	Dezimalzahl
F	F	F	F	F	T	000001	1
F	F	T	F	F	T	001001	9
F	T	F	T	F	T	010101	21
T	T	T	T	F	F	111100	60

FPGA

„FPGA Main.vi“ ist state machine:

FPGA

„FPGA Main.vi“ ist state machine:

- safe state

FPGA

„FPGA Main.vi“ ist state machine:

- safe state
- manual

FPGA

„FPGA Main.vi“ ist state machine:

- safe state
- manual
- control

FPGA

„FPGA Main.vi“ ist state machine:

- safe state
- manual
- control
  - configuration

FPGA

„FPGA Main.vi“ ist state machine:

- safe state
- manual
- control
  - configuration
  - ready to run

FPGA

„FPGA Main.vi“ ist state machine:

- safe state
- manual
- control
  - configuration
  - ready to run
  - run

FPGA

FPGA

> Zustand von Frontpanel-Objekt abhängig

FPGA

- Zustand von Frontpanel-Objekt abhängig
- Änderungen durch RT über Read/Write Control

FPGA

- Zustand von Frontpanel-Objekt abhängig
- Änderungen durch RT über Read/Write Control
- Werte und Einstellungen des Reaktors über FPGA I/O Nodes

FPGA

- Zustand von Frontpanel-Objekt abhängig
- Änderungen durch RT über Read/Write Control
- Werte und Einstellungen des Reaktors über FPGA I/O Nodes
- **Keine Optimierung**

FPGA - Reaktor

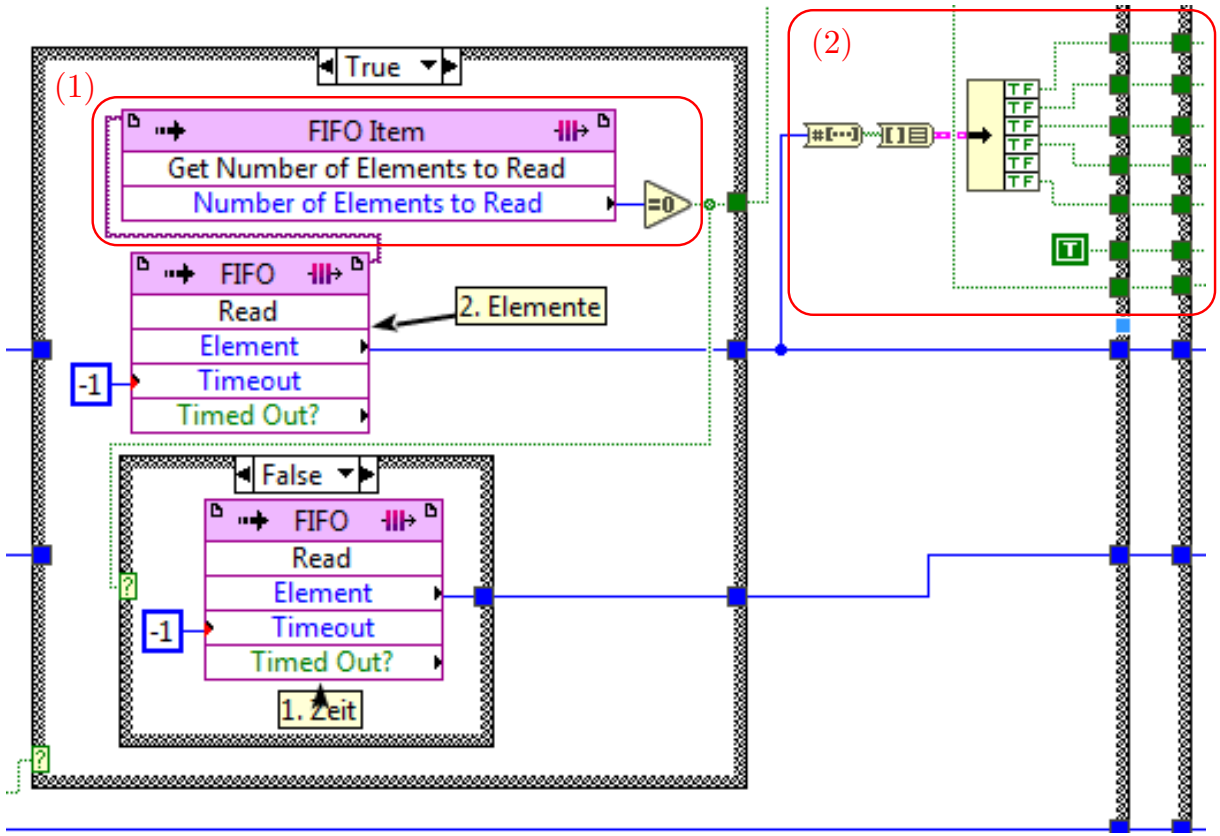


Abbildung: FPGA Main.vi

FPGA - Reaktor

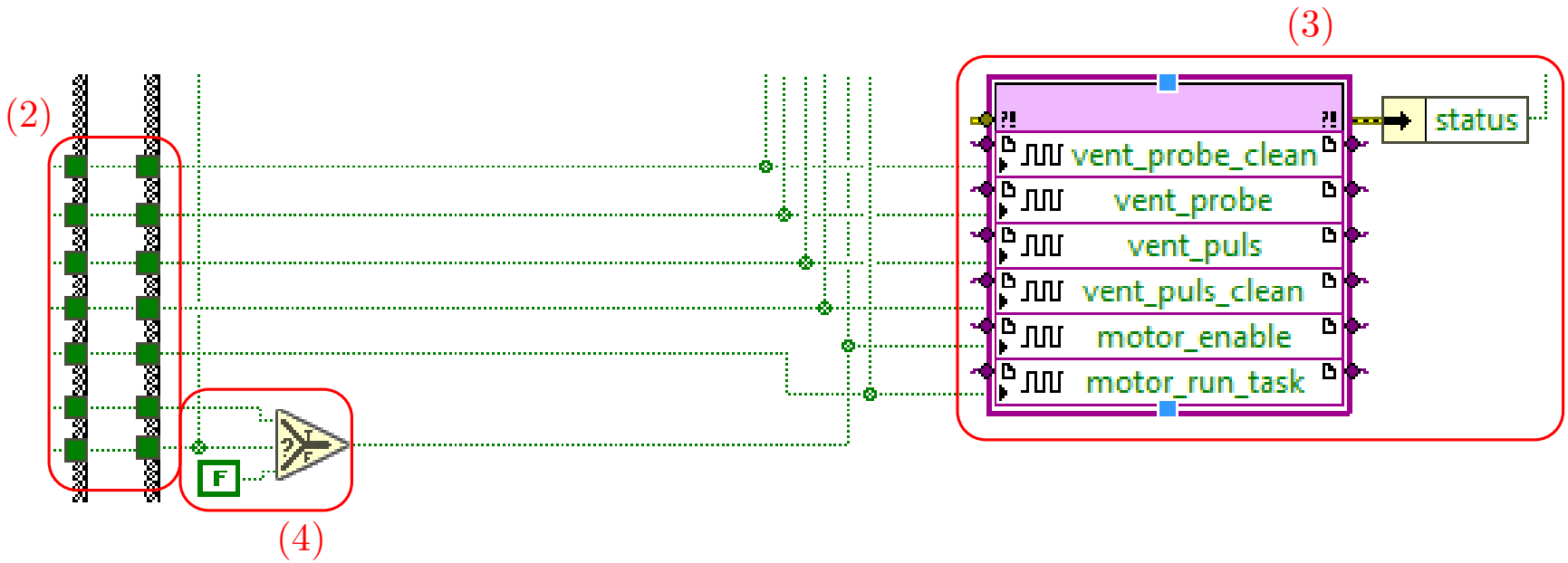
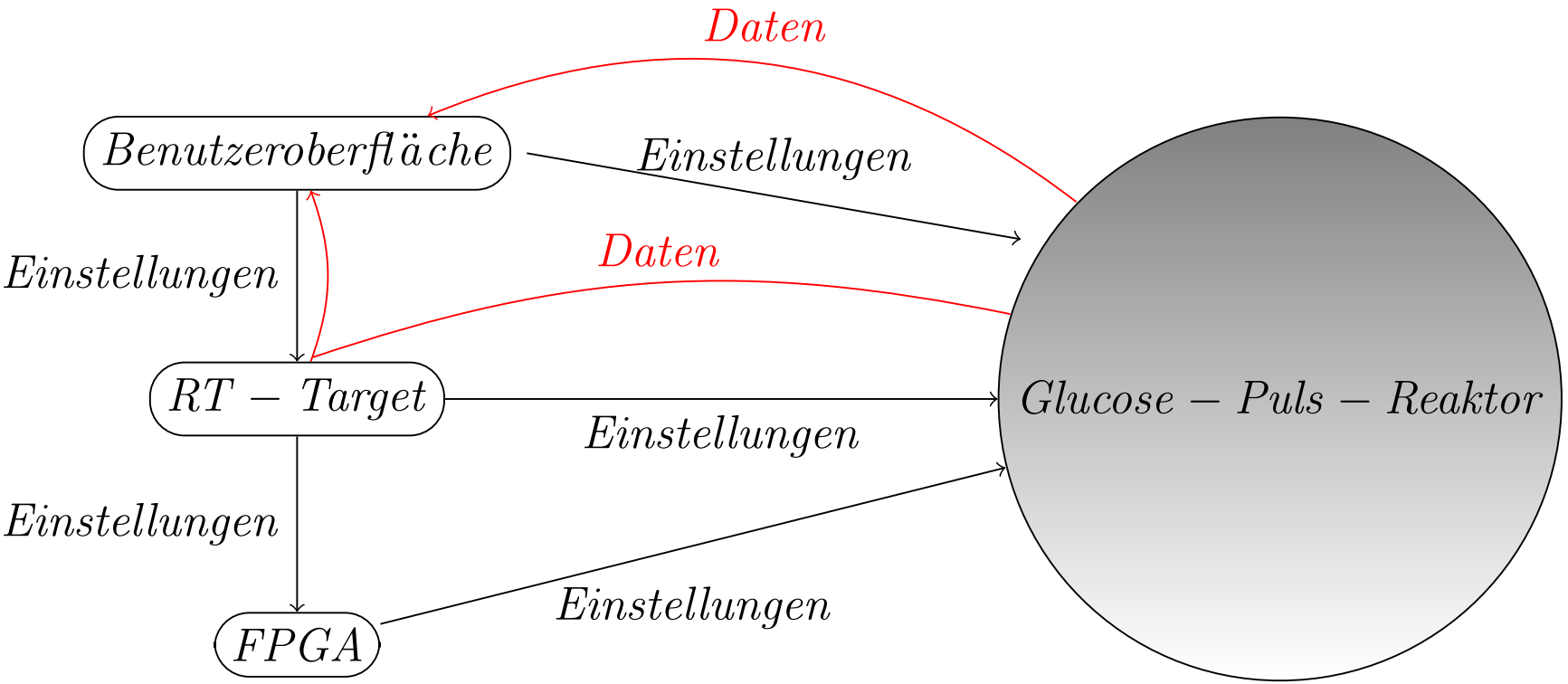


Abbildung: FPGA Main.vi

Glucose-Puls-Reaktor-Programm



Kommunikation von der Benutzeroberfläche aus

> Umgebungsvariablen zur direkten Kommunikation

Kommunikation von der Benutzeroberfläche aus

> Umgebungsvariablen zur direkten Kommunikation

> mit physikalisch vorhandenen Ventilen verbunden

Kommunikation von der Benutzeroberfläche aus

- Umgebungsvariablen zur direkten Kommunikation
- mit physikalisch vorhandenen Ventilen verbunden
- Einstellungen, die Benutzer vornimmt an Reaktor übertragen

Kommunikation von der Benutzeroberfläche aus

- Umgebungsvariablen zur direkten Kommunikation
- mit physikalisch vorhandenen Ventilen verbunden
- Einstellungen, die Benutzer vornimmt an Reaktor übertragen
- Werte auf Benutzeroberfläche ausgegeben

Kommunikation von der Benutzeroberfläche aus

- Umgebungsvariablen zur direkten Kommunikation
- mit physikalisch vorhandenen Ventilen verbunden
- Einstellungen, die Benutzer vornimmt an Reaktor übertragen
- Werte auf Benutzeroberfläche ausgegeben
- RT => Stream => Queue => Benutzeroberfläche

Kommunikation von der Benutzeroberfläche aus

- Umgebungsvariablen zur direkten Kommunikation
- mit physikalisch vorhandenen Ventilen verbunden
- Einstellungen, die Benutzer vornimmt an Reaktor übertragen
- Werte auf Benutzeroberfläche ausgegeben
- RT => Stream => Queue => Benutzeroberfläche
- **Optimierung: Auskommentieren nicht verwendeter Werte**

Umgebungsvariablen

Abbildung: Nicht verwendete Daten

Umgebungsvariablen

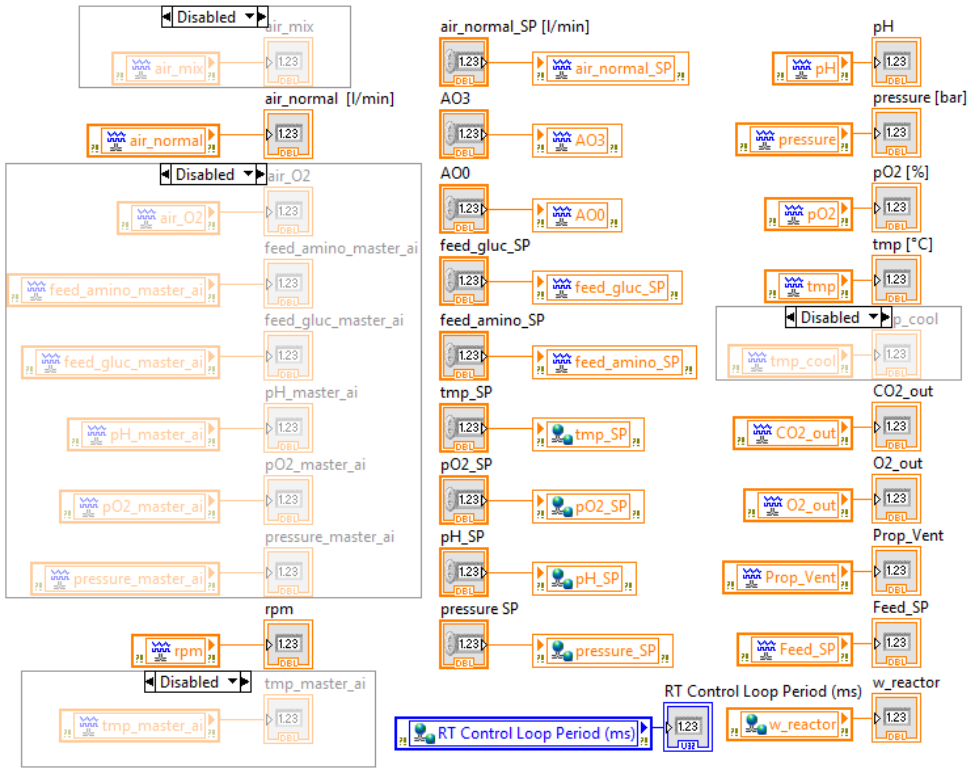


Abbildung: Auskommentierte Umgebungsvariablen

Kommunikation vom RT-Target aus

> Umgebungsvariablen zur direkten Kommunikation

Kommunikation vom RT-Target aus

> Umgebungsvariablen zur direkten Kommunikation
> Steuerung der Vent-Inoc Ventile

Kommunikation vom RT-Target aus

- Umgebungsvariablen zur direkten Kommunikation
- Steuerung der Vent-Inoc Ventile
- Sechs Zugriffe

Kommunikation vom RT-Target aus

- Umgebungsvariablen zur direkten Kommunikation
- Steuerung der Vent-Inoc Ventile
- Sechs Zugriffe
- **Keine Optimierung**

Ergebnis der Analyse

Tabelle: Verwendung der verschiedenen Datenübertragungstypen im Glucose-Puls-Reaktor-Programm

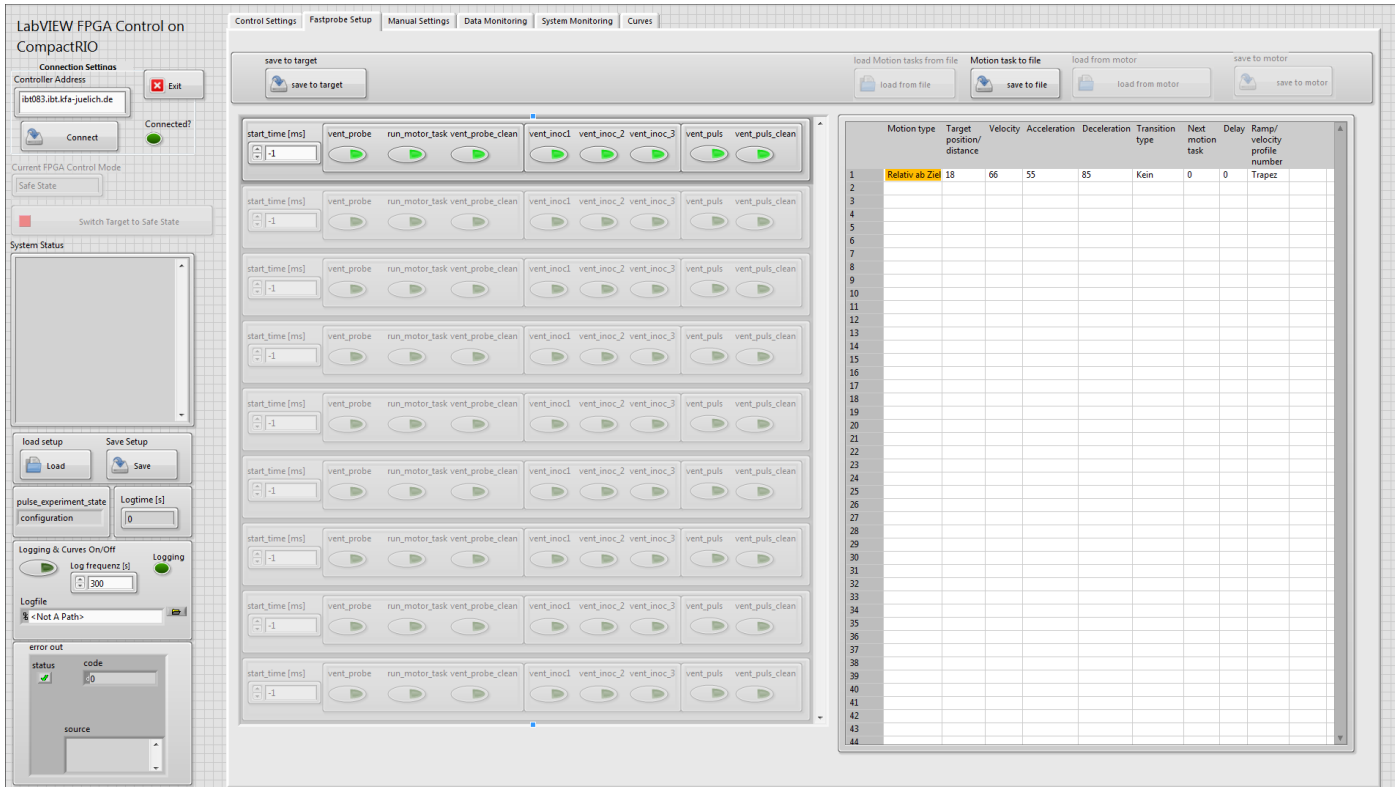
Typ	B	B-T	T	T-F	F	T-R	F-R	B-R
Lokale	42	0	0	0	0	0	0	0
Globale (Z)	20	3	22	0	0	0	0	0
Funk. globale (Z)	6	0	0	0	0	0	0	0
Umgebungs. (Z)	5	12	0	0	0	6	0	40
E.Knoten	68	0	3	0	0	0	0	0
M.Knoten	0	0	0	1	0	0	0	0
R./W. Control	0	0	0	4	0	0	0	0
FIFO	0	0	0	1	0	0	0	0
Queue	1	0	1	0	0	0	0	0
Stream	0	2	0	0	0	0	0	0
FPGA I/O Node	0	0	0	0	0	0	2	0

Ergebnis der Analyse

Tabelle: Verwendung der verschiedenen Datenübertragungstypen im Glucose-Puls-Reaktor-Programm – nach erster Optimierung

Typ	B	B-T	T	T-F	F	T-R	F-R	B-R
Lokale	42	0	0	0	0	0	0	0
Globale (Z)	20	3	22	0	0	0	0	0
Funk. globale (Z)	6	0	0	0	0	0	0	0
Umgebungs. (Z)	5	12	0	0	0	6	0	31
E.Knoten	68	0	3	0	0	0	0	0
M.Knoten	0	0	0	1	0	0	0	0
R./W. Control	0	0	0	4	0	0	0	0
FIFO	0	0	0	1	0	0	0	0
Queue	1	0	1	0	0	0	0	0
Stream	0	2	0	0	0	0	0	0
FPGA I/O Node	0	0	0	0	0	0	2	0

Benutzeroberfläche



Vor der Optimierung

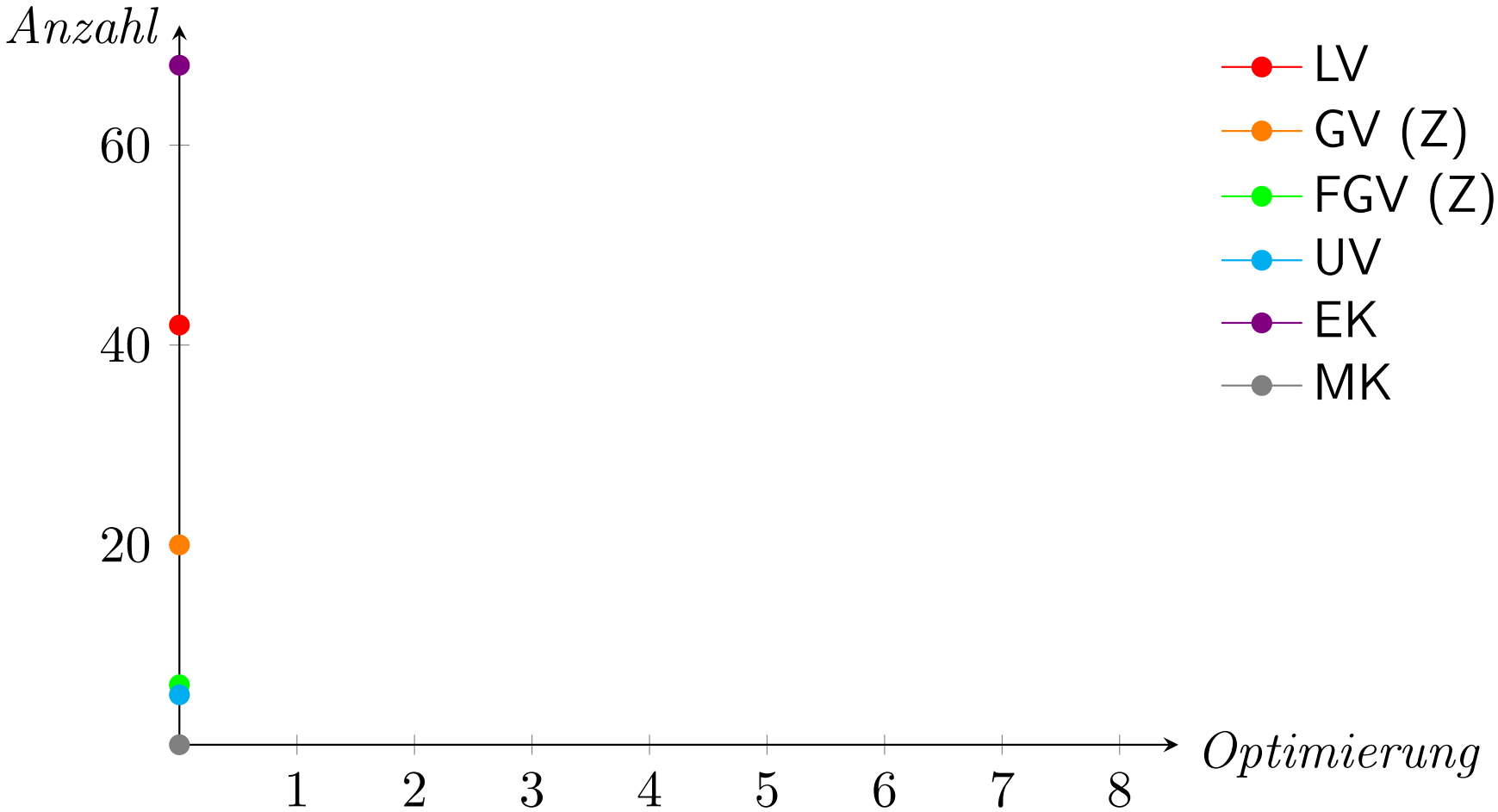


Abbildung: Darstellung des Optimierungsprozesses–Schritt 0

Lokale Variable „Connected?“

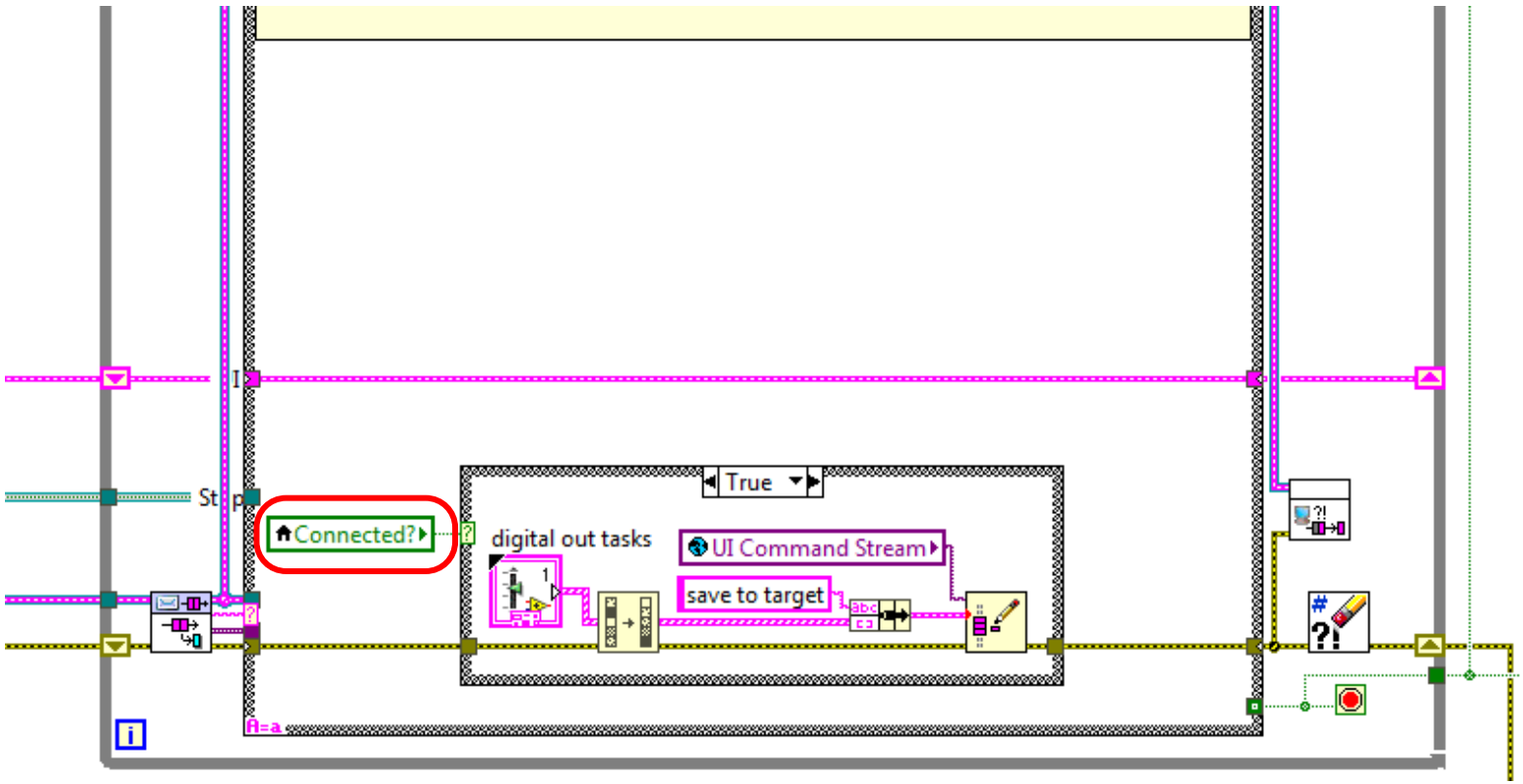


Abbildung: Vorher: Verwendung der lokalen Variablen „Connected?“

Lokale Variable „Connected?“

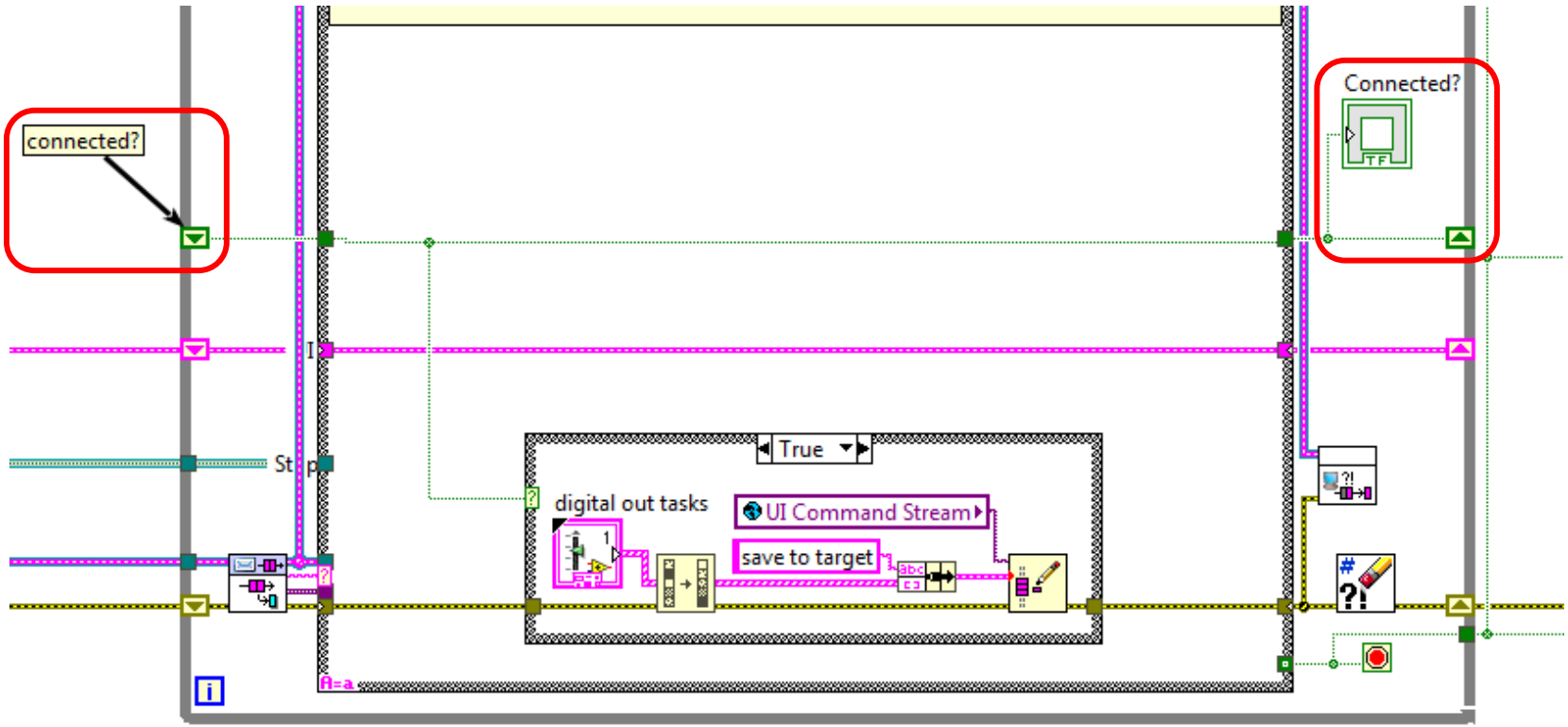


Abbildung: Nachher: Verwendung des Shift-Registers statt „Connected?“

Lokale Variable „Connected?“

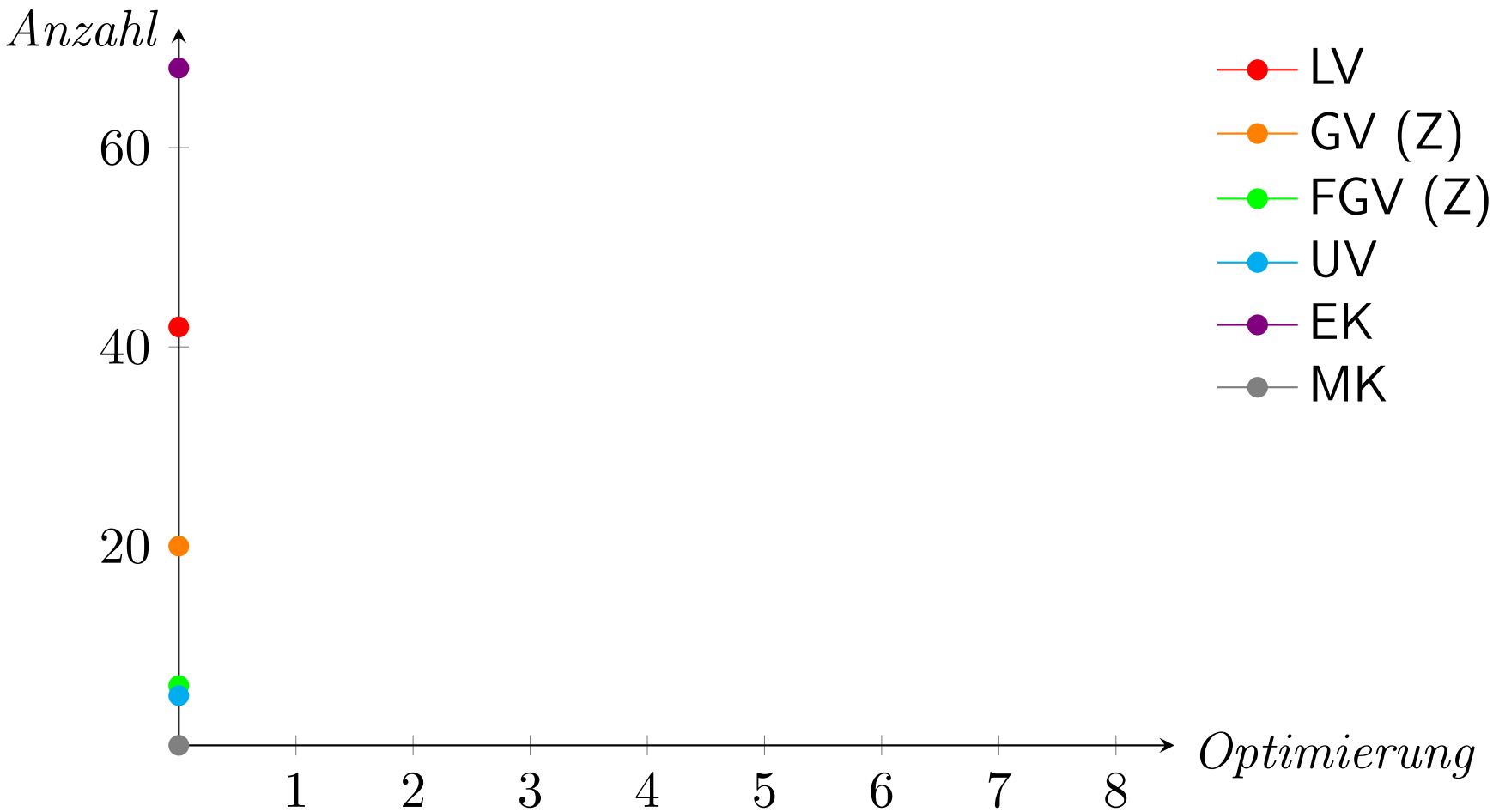


Abbildung: Darstellung des Optimierungsprozesses–Schritt 0

Lokale Variable „Connected?“

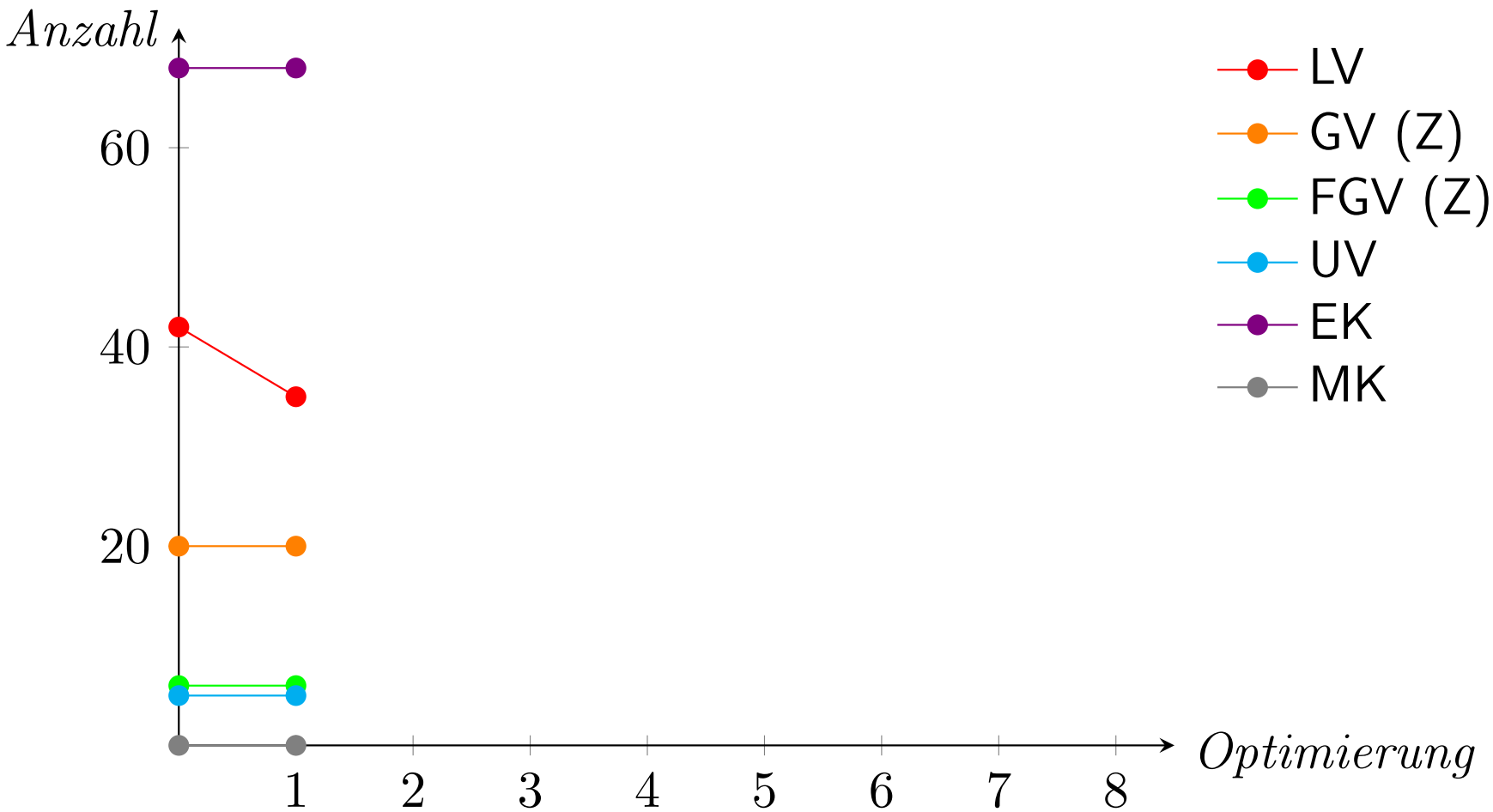


Abbildung: Darstellung des Optimierungsprozesses–Schritt 1

Lokale Variablen des Schließungsfensters

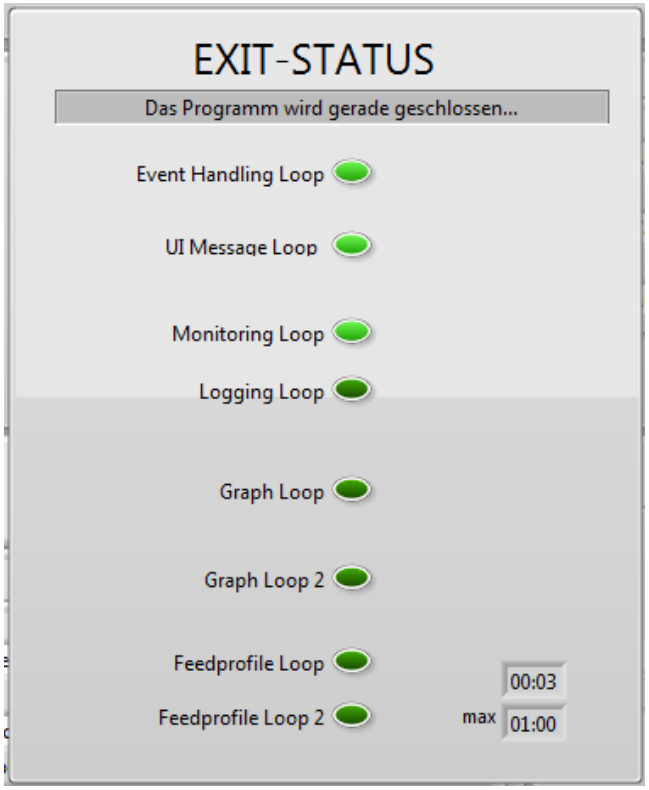


Abbildung: Anzeige des Schließungsfensters bei Beenden des Programms

Lokale Variablen des Schließungsfensters

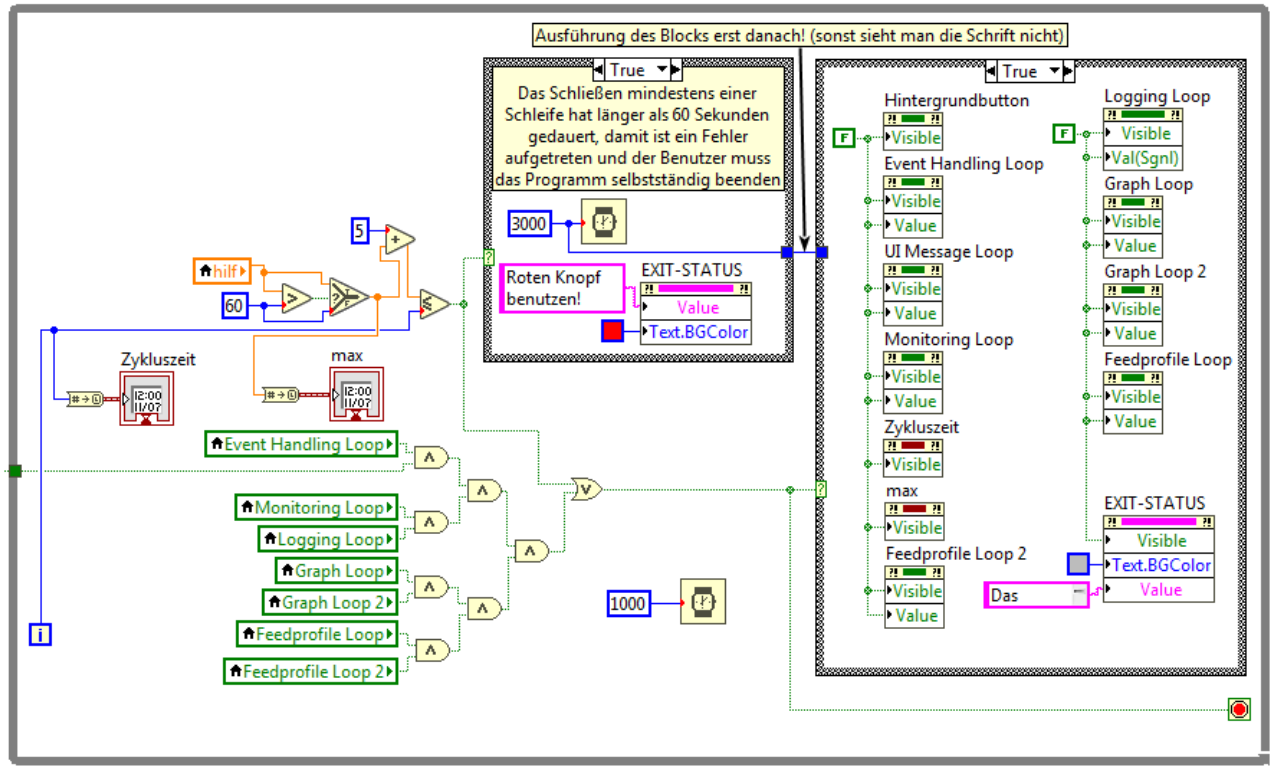


Abbildung: Überprüfung der Zustände vorher

Lokale Variablen des Schließungsfensters

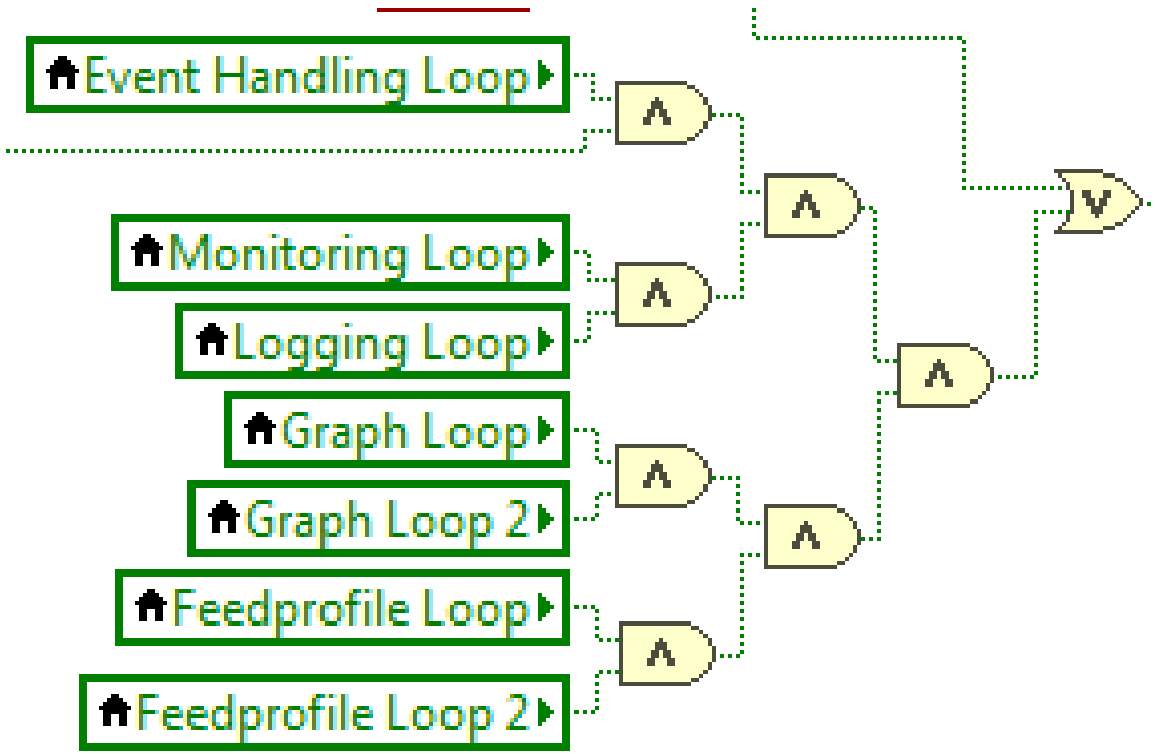


Abbildung: Überprüfung der Zustände vorher

Lokale Variablen des Schließungsfensters

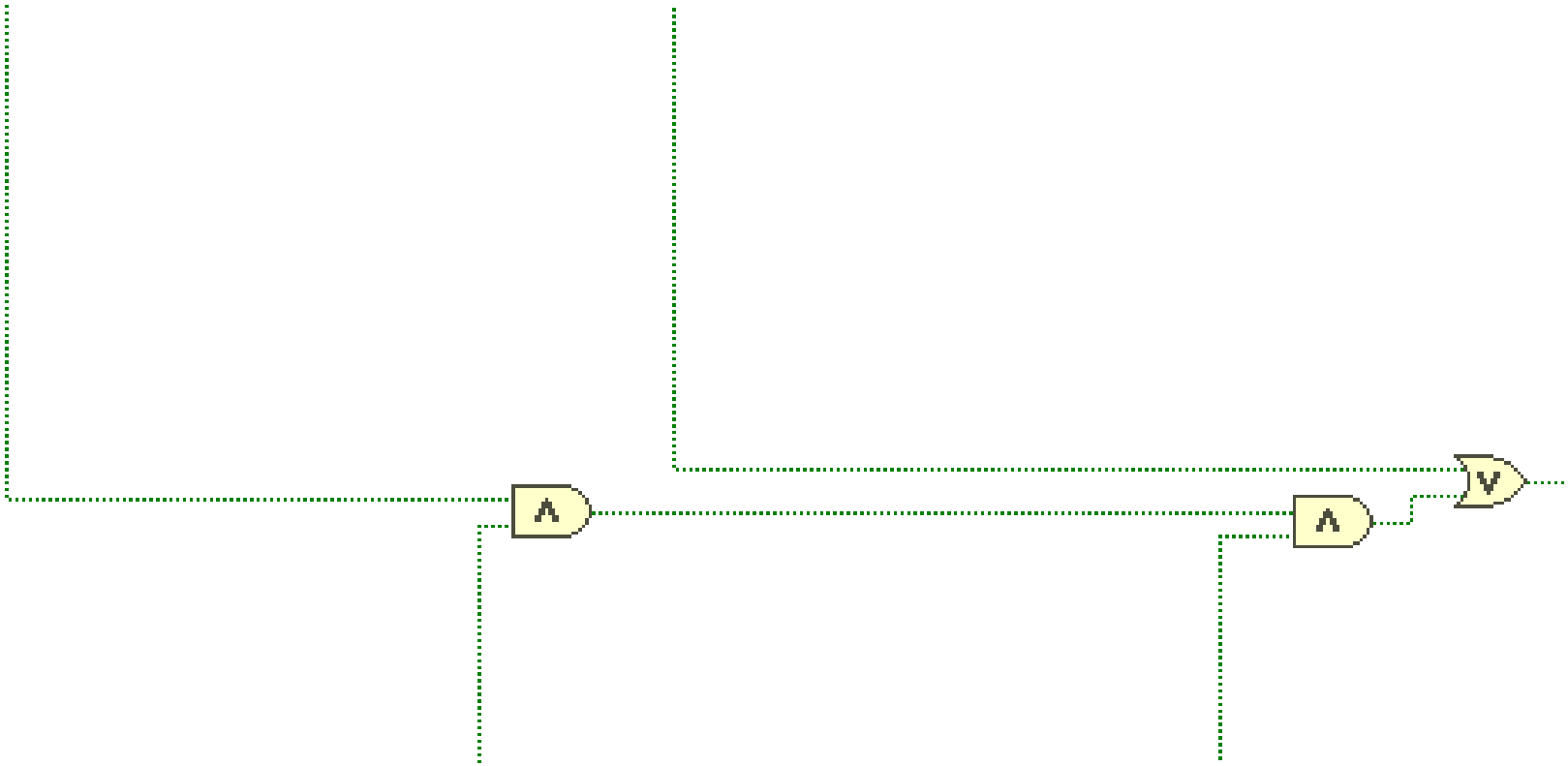


Abbildung: Überprüfung der Zustände Nachher (ohne lokale Variablen)

Lokale Variablen des Schließungsfensters

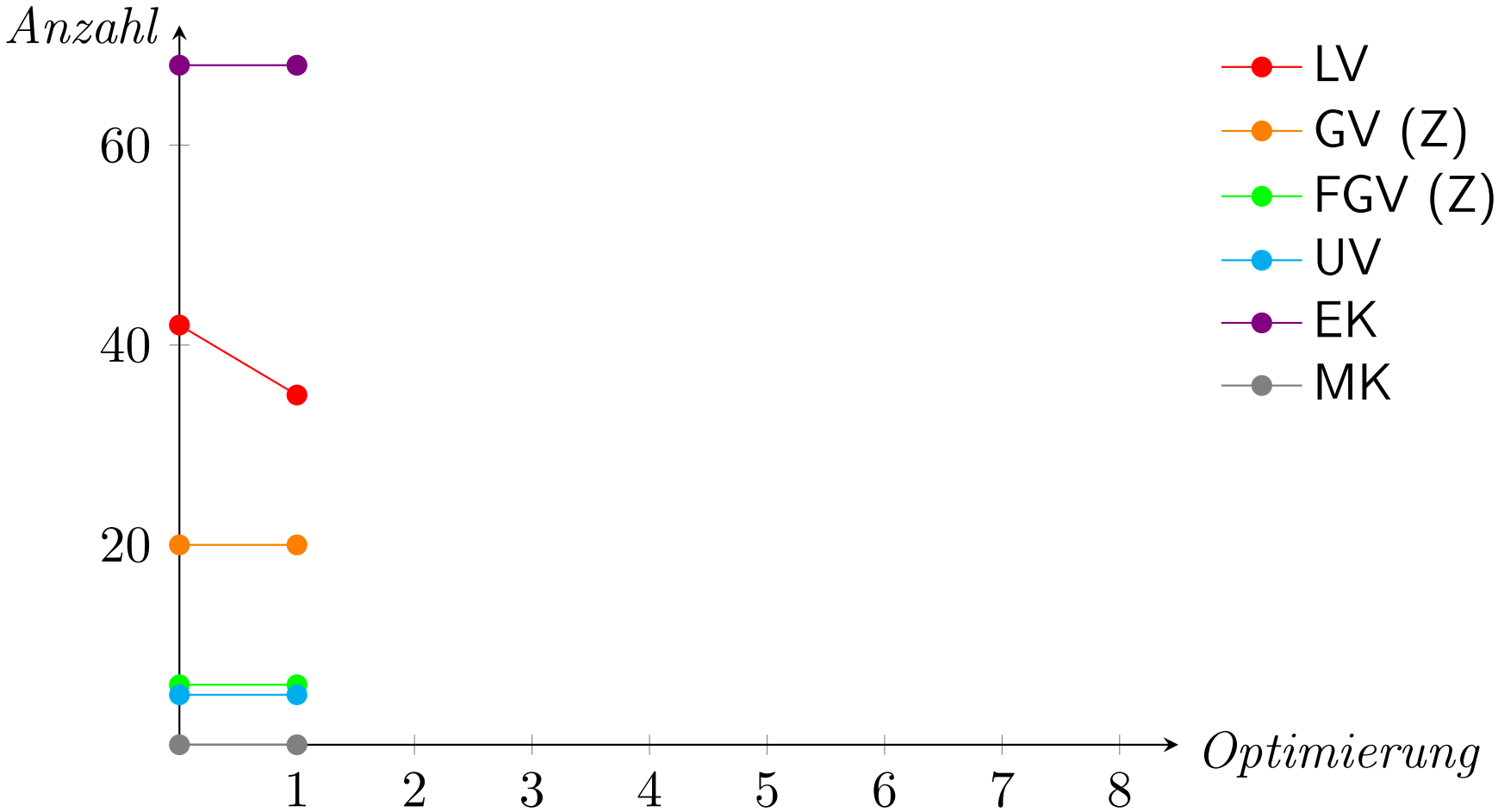


Abbildung: Darstellung des Optimierungsprozesses–Schritt 1

Lokale Variablen des Schließungsfensters

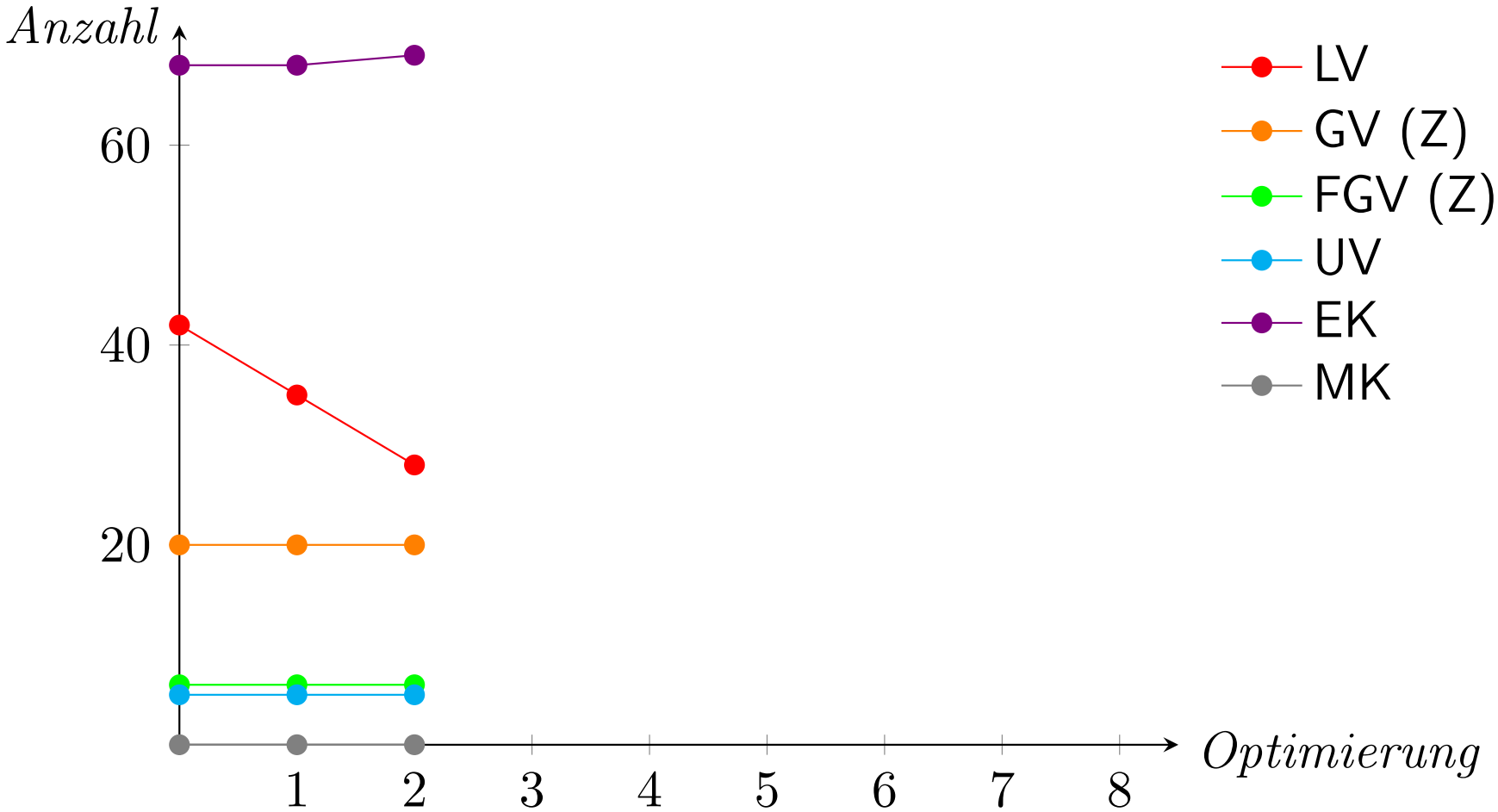


Abbildung: Darstellung des Optimierungsprozesses–Schritt 2

Globale Variable „All UI Loop Stop“

Globale Variable „All UI Loop Stop“

> 10 Verwendungen => nur in „UI Main.vi“

Globale Variable „All UI Loop Stop“

- 10 Verwendungen => nur in „UI Main.vi“
- Umwandlung in lokale Variablen

Globale Variable „All UI Loop Stop“

- 10 Verwendungen => nur in „UI Main.vi“
- Umwandlung in lokale Variablen
- Keine Änderung aus anderen VIs möglich

Globale Variable „All UI Loop Stop“

- 10 Verwendungen => nur in „UI Main.vi“
- Umwandlung in lokale Variablen
- Keine Änderung aus anderen VIs möglich
- - 10 Zugriffe auf globale Variable „All UI Loop Stop“

Globale Variable „All UI Loop Stop“

- 10 Verwendungen => nur in „UI Main.vi“
- Umwandlung in lokale Variablen
- Keine Änderung aus anderen VIs möglich
- - 10 Zugriffe auf globale Variable „All UI Loop Stop“
- + 9 lokale Variablen „All UI Loop Stop“

Globale Variable „All UI Loop Stop“

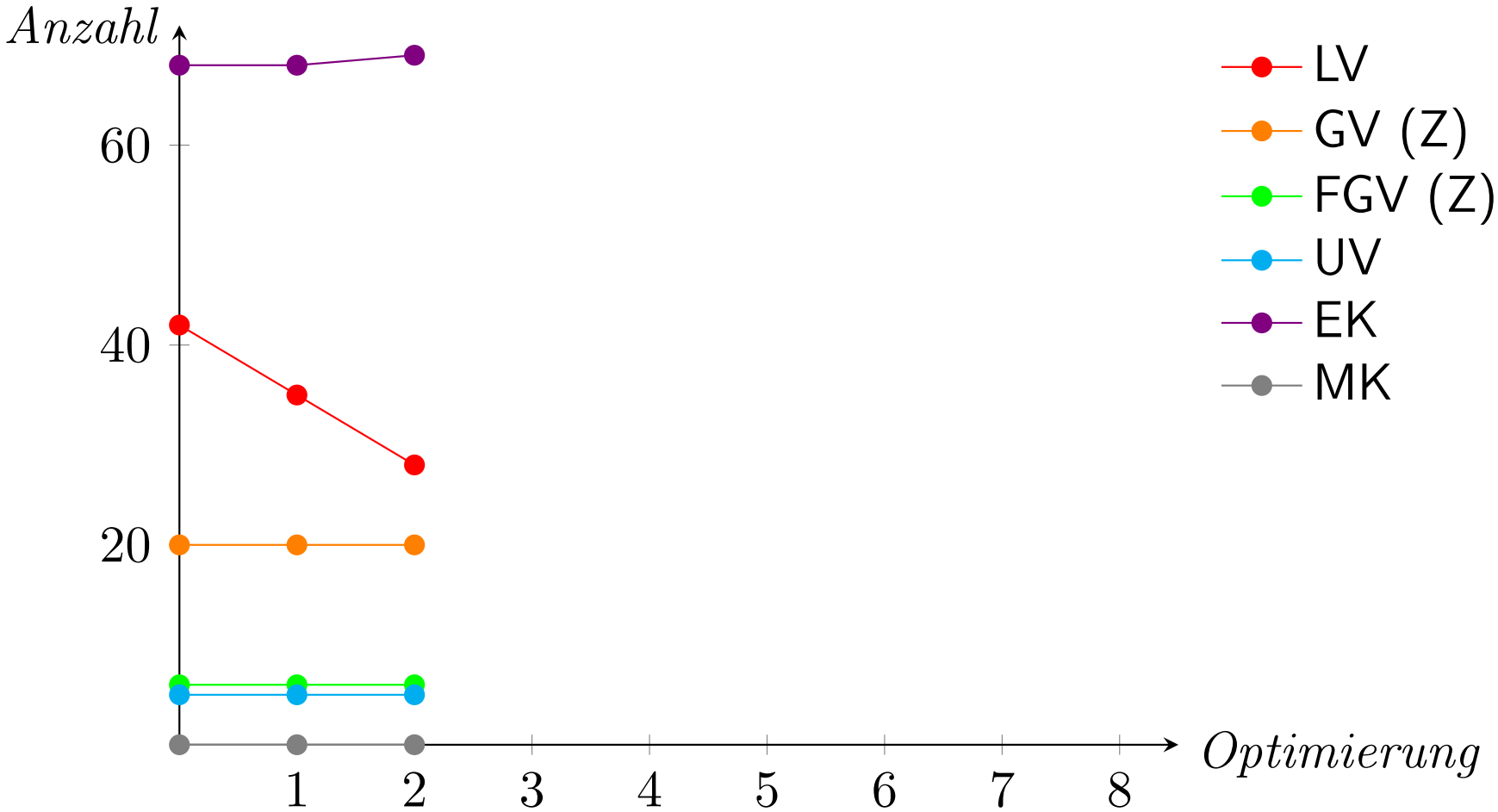


Abbildung: Darstellung des Optimierungsprozesses–Schritt 2

Globale Variable „All UI Loop Stop“

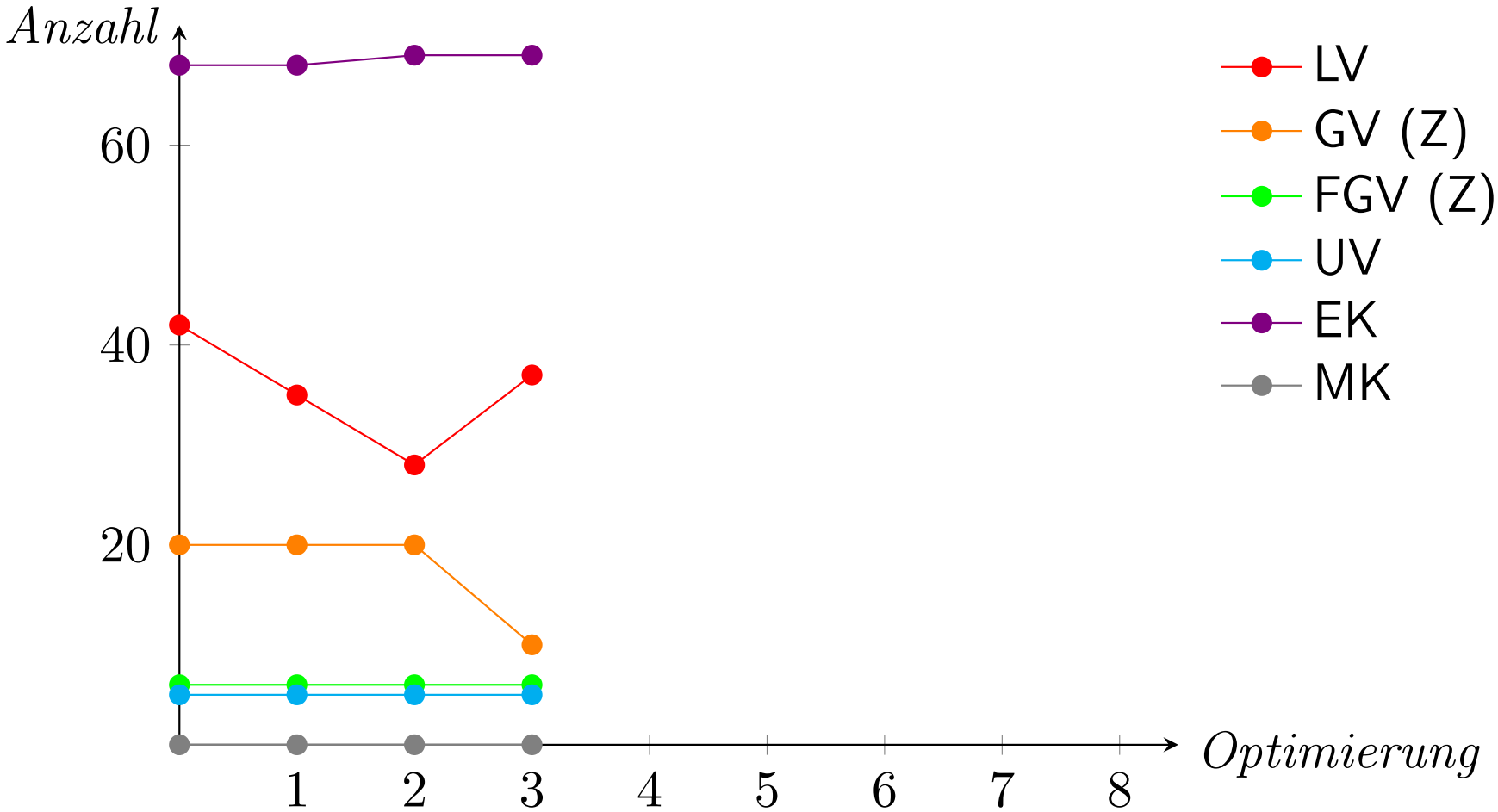


Abbildung: Darstellung des Optimierungsprozesses–Schritt 3

Globale Variable „UI Command Stream“

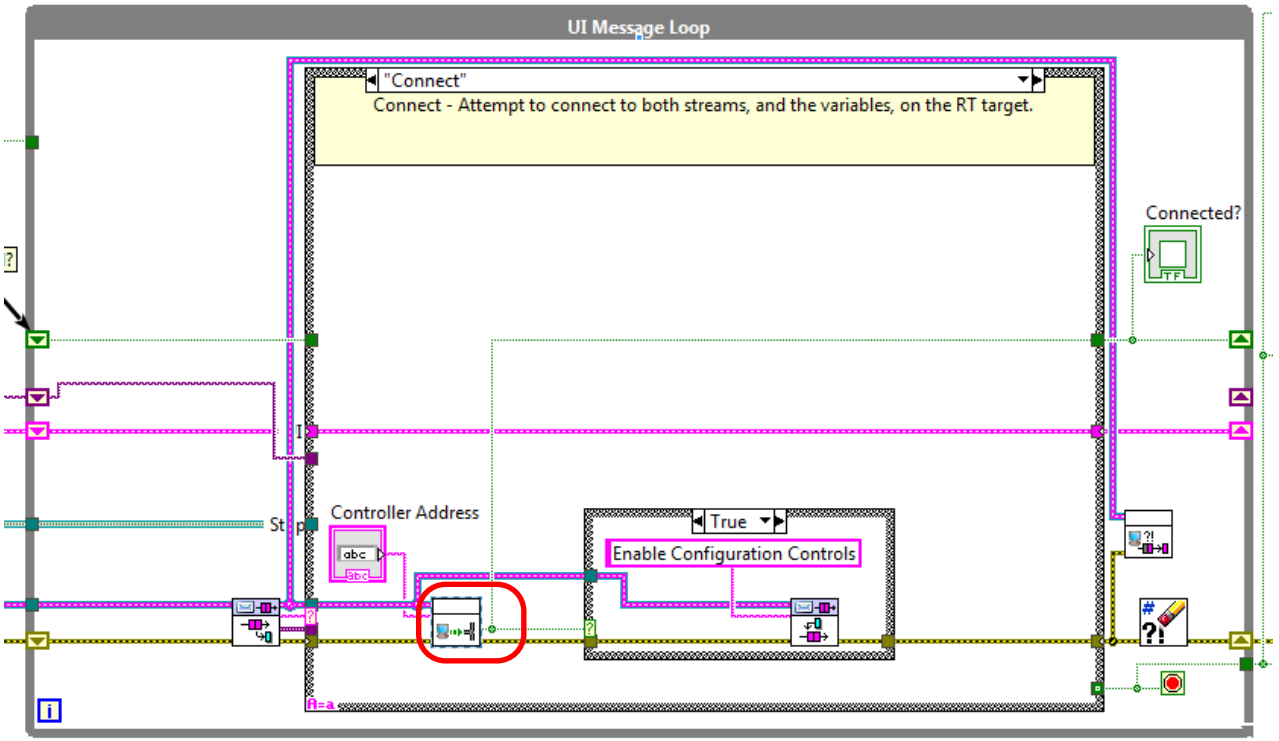


Abbildung: Vorher: Keine Rückgabe des UI - Initiate Connection.vi

Globale Variable „UI Command Stream“

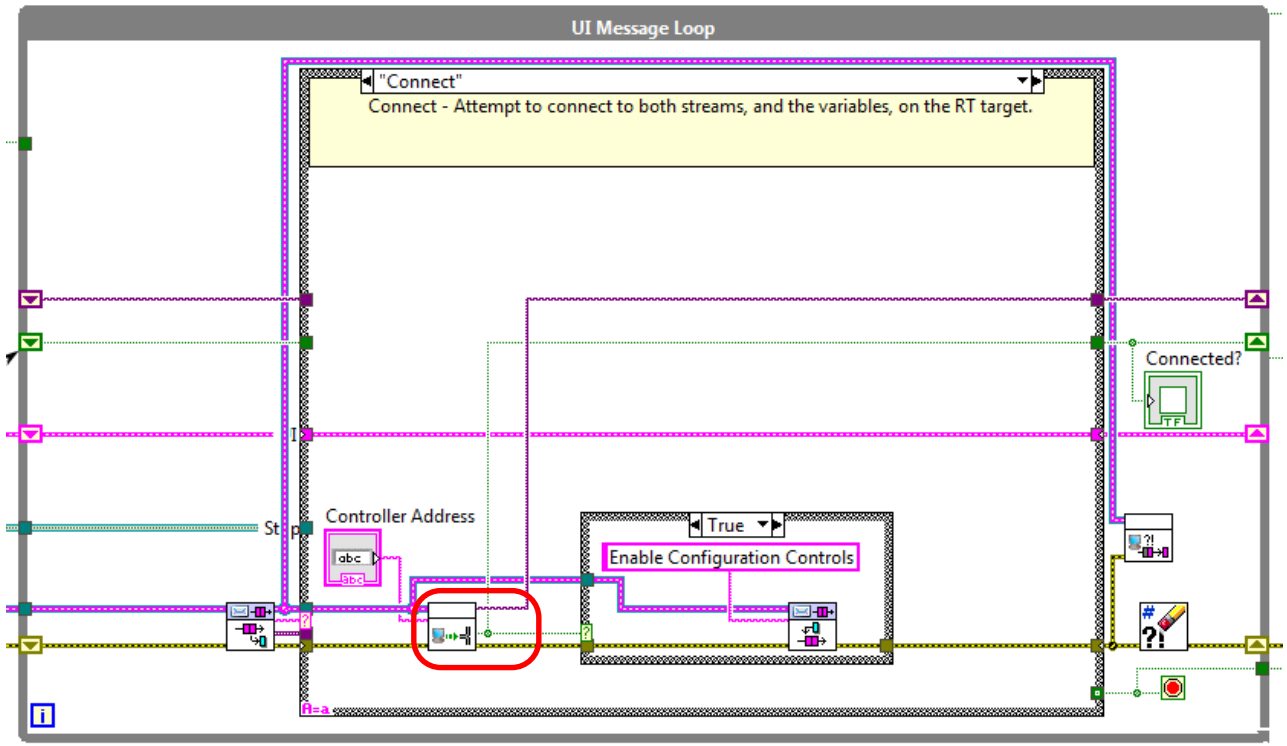


Abbildung: Nachher: Rückgabe durch UI - Initiate Connection.vi

Globale Variable „UI Command Stream“

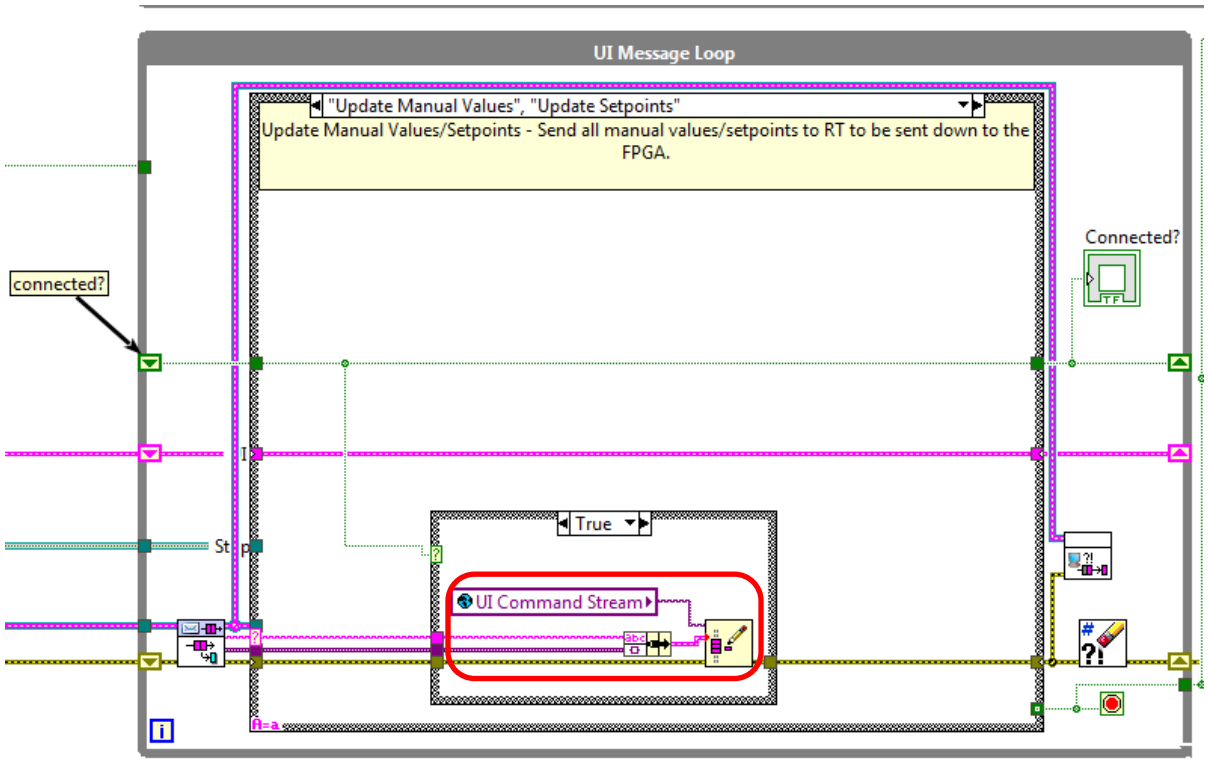


Abbildung: Vorher: Verwendung der globalen Variable

Globale Variable „UI Command Stream“

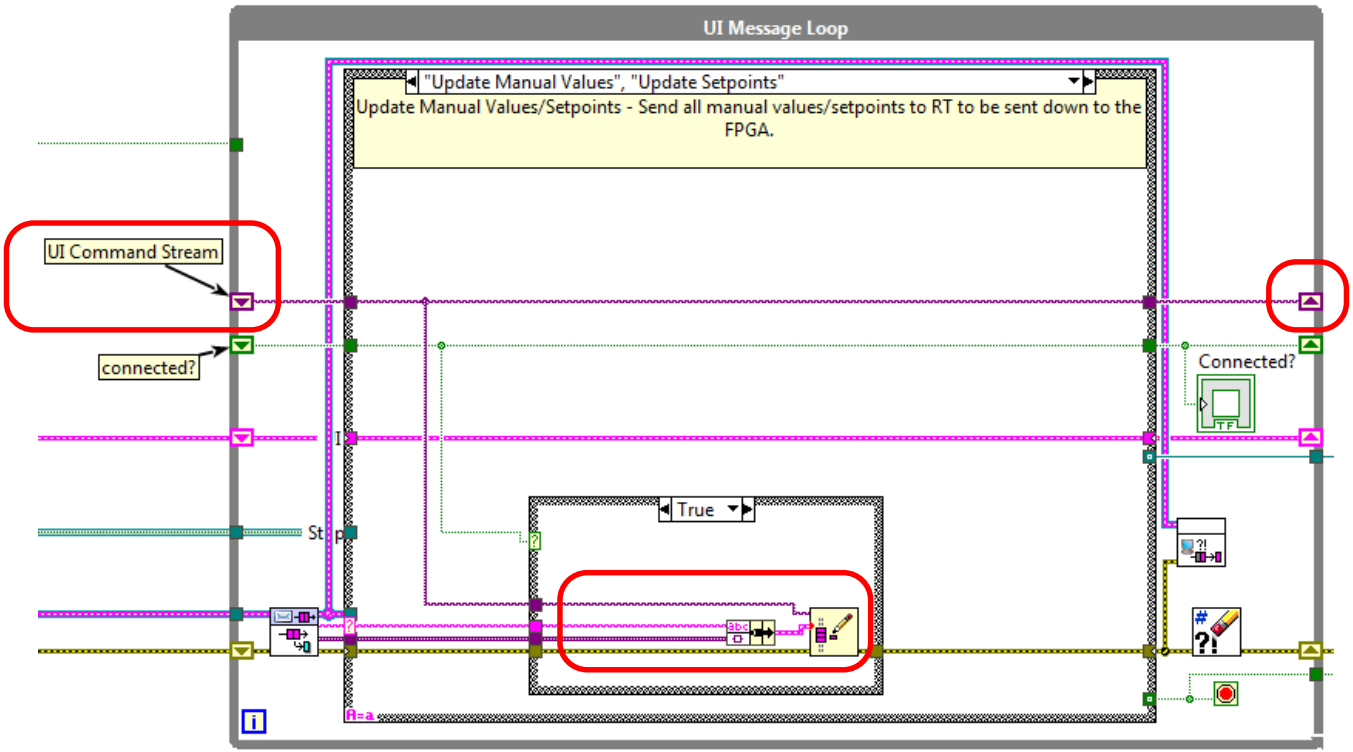


Abbildung: Nachher: Verwendung des Schieberegisters

Globale Variable „UI Command Stream“

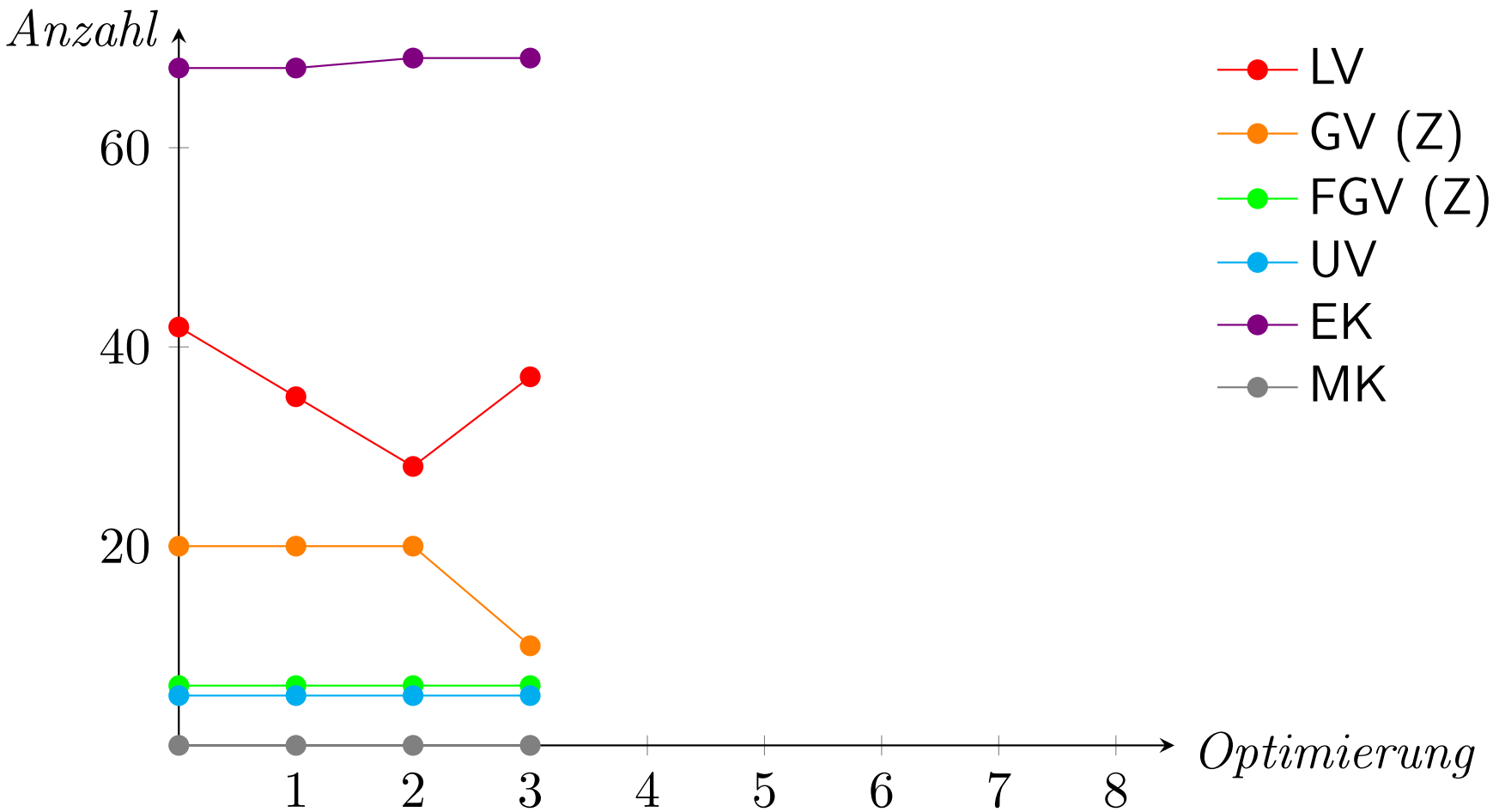


Abbildung: Darstellung des Optimierungsprozesses–Schritt 3

Globale Variable „UI Command Stream“

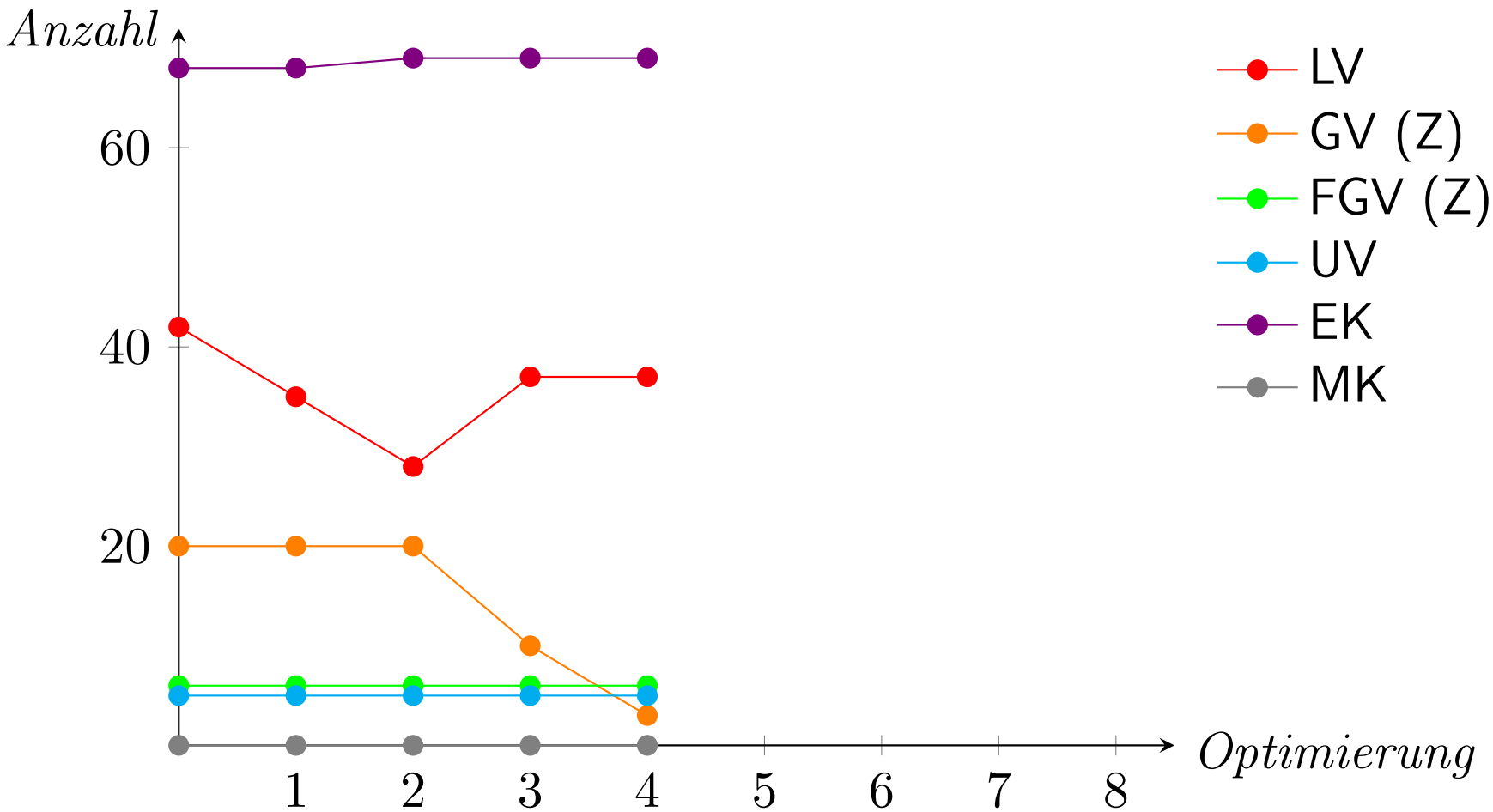


Abbildung: Darstellung des Optimierungsprozesses–Schritt 4

Eigenschaftsknoten des Startfensters

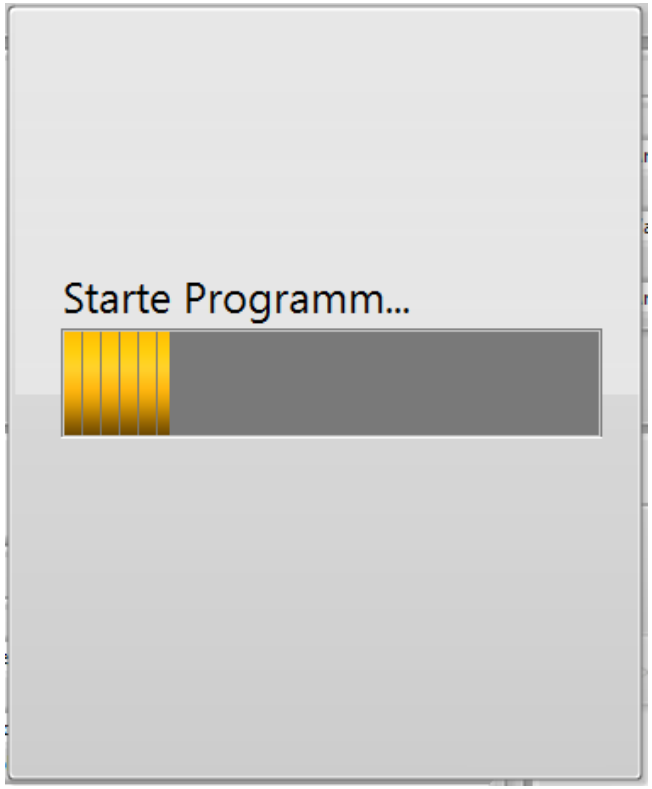


Abbildung: Anzeige des Startfensters beim Starten des Programms

Eigenschaftsknoten des Startfensters

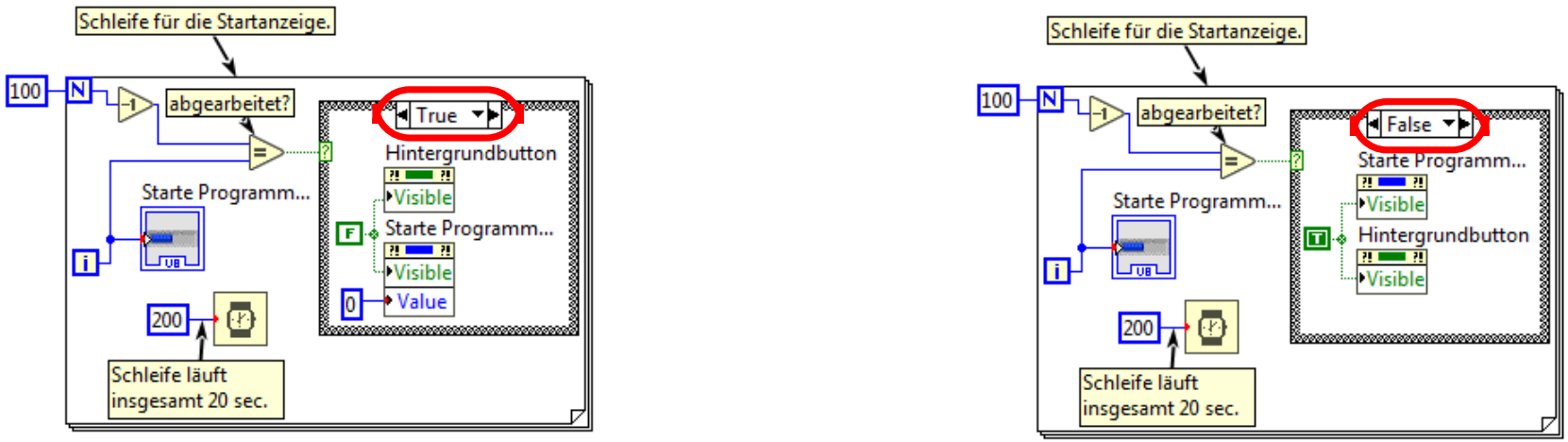


Abbildung: Programmierung der Startanzeige vorher

Eigenschaftsknoten des Startfensters

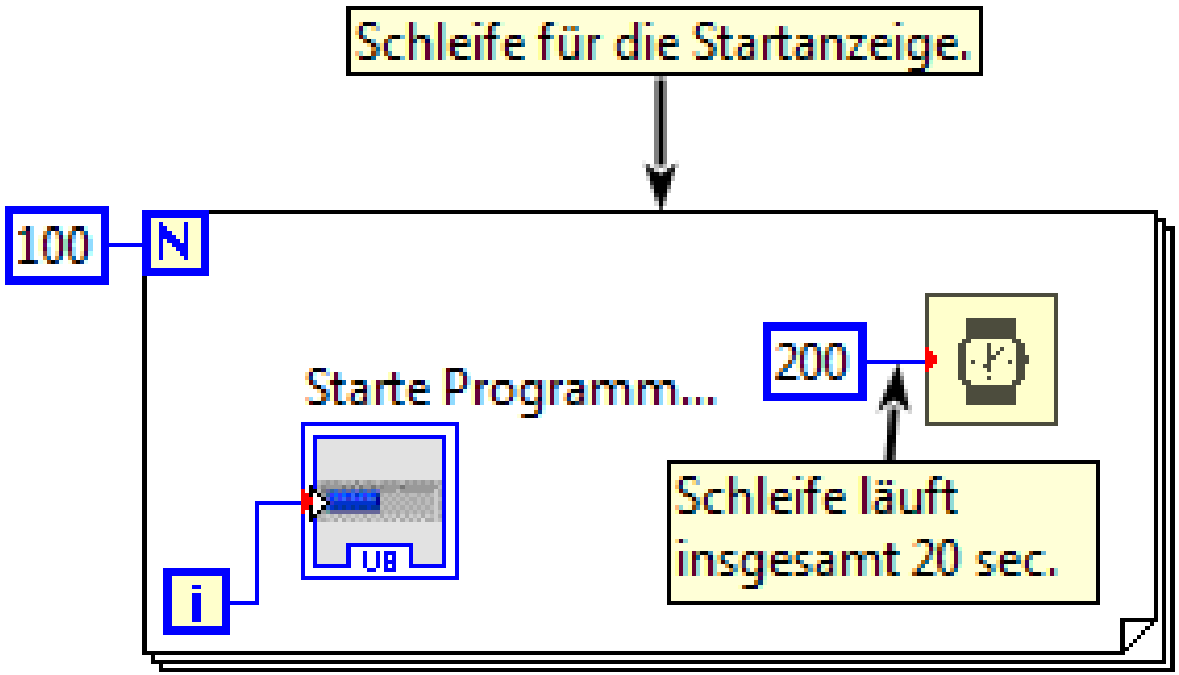


Abbildung: Auslagerung der Startanzeige in ein SubVI

Eigenschaftsknoten des Startfensters

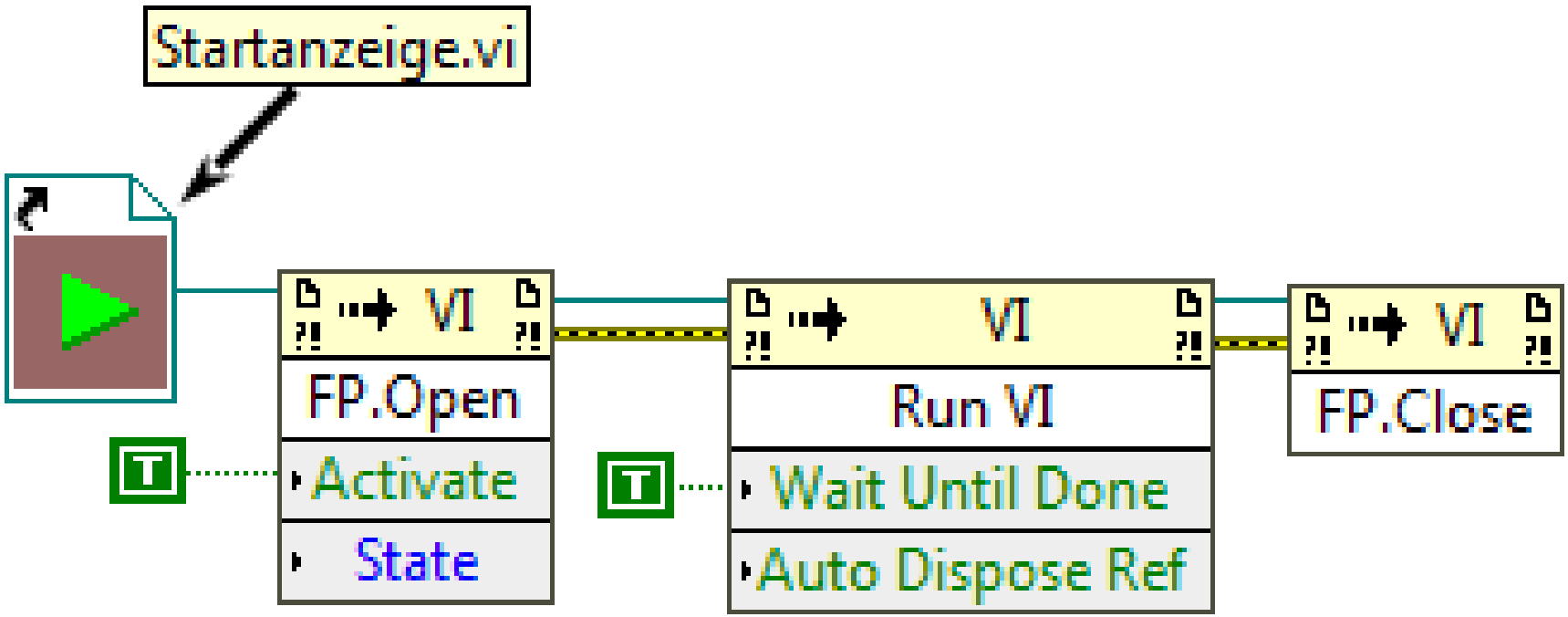


Abbildung: Aufruf des Startanzeige-VI

Eigenschaftsknoten des Startfensters

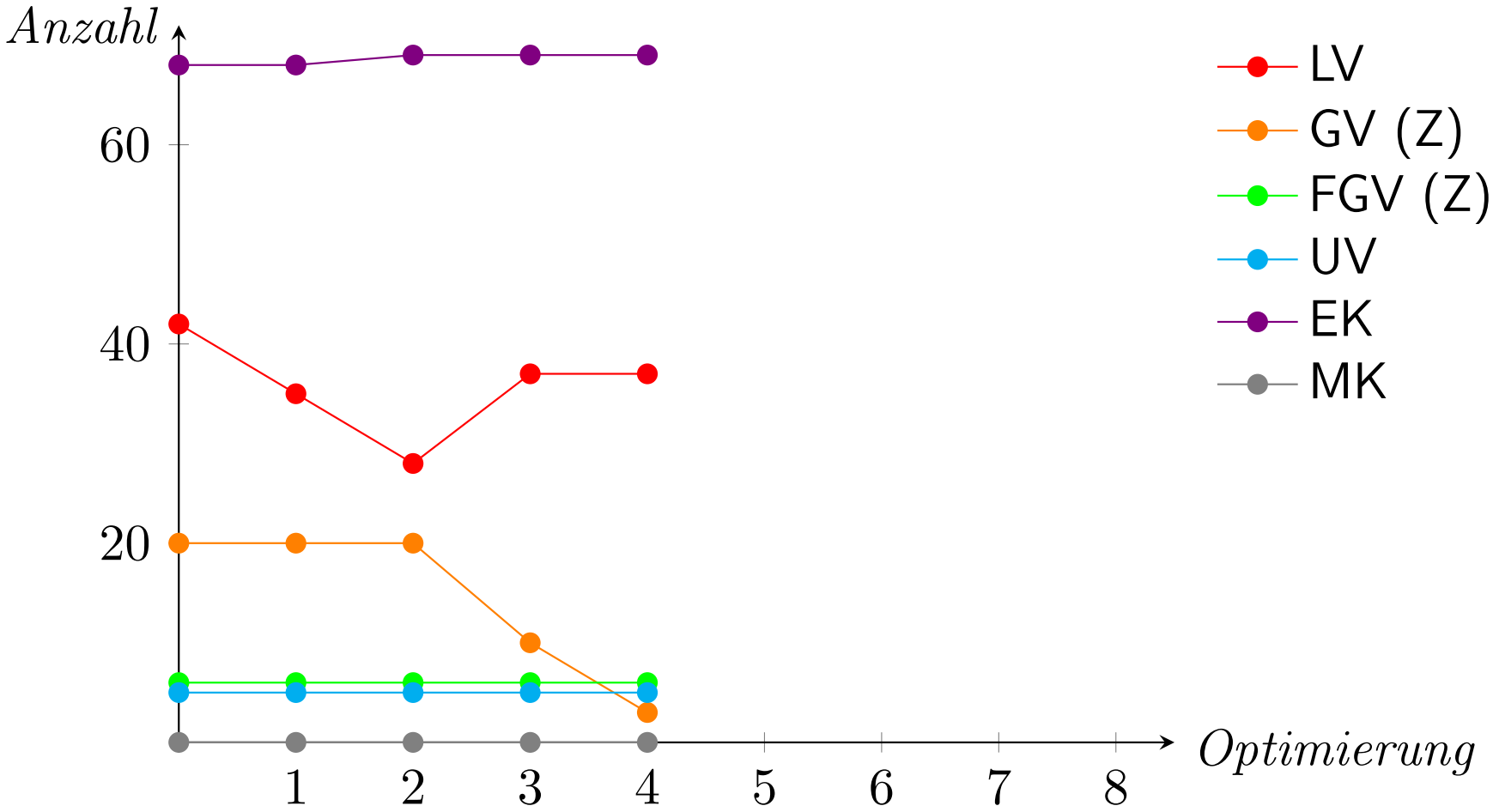


Abbildung: Darstellung des Optimierungsprozesses–Schritt 4

Eigenschaftsknoten des Startfensters

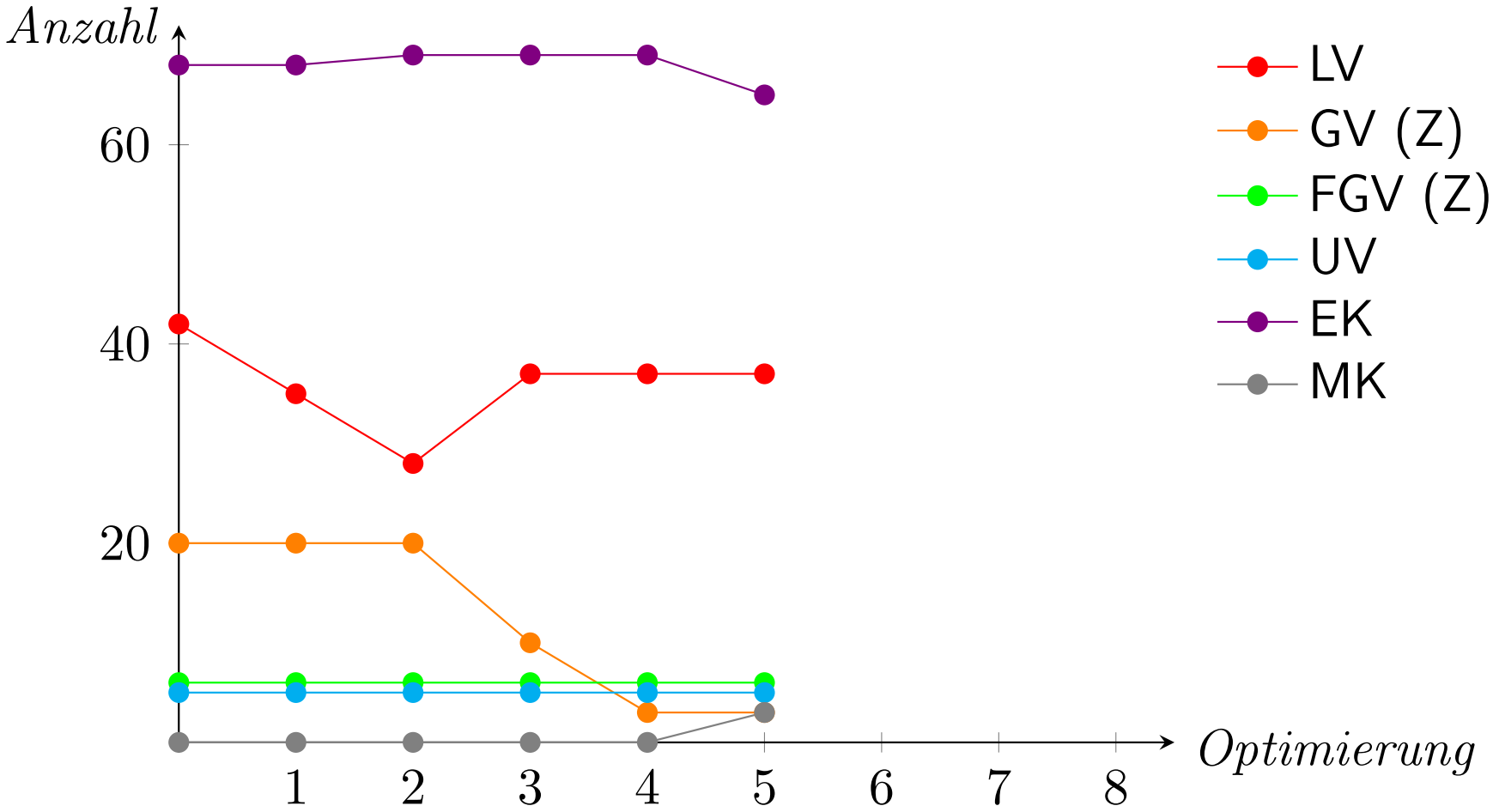


Abbildung: Darstellung des Optimierungsprozesses–Schritt 5

Eigenschaftsknoten des Schließungsfensters

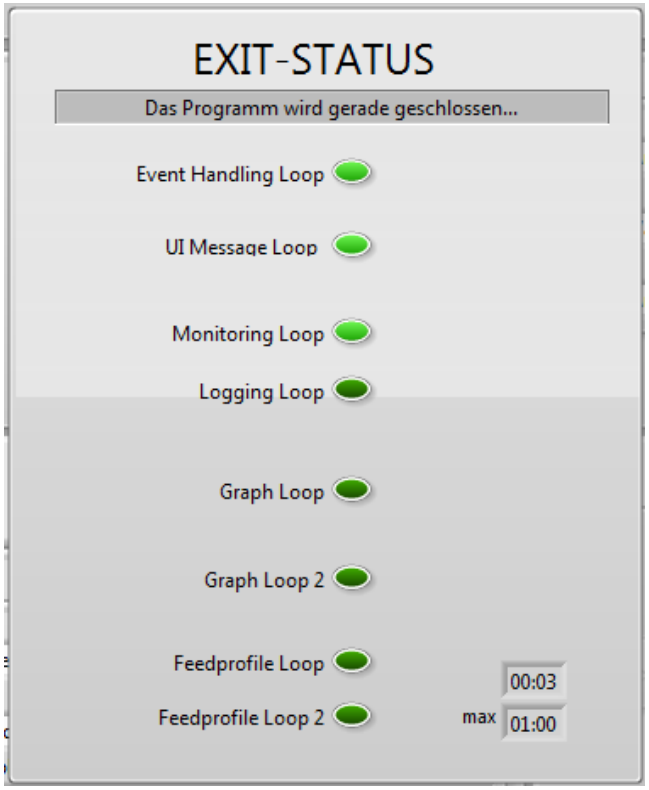


Abbildung: Anzeige des Schließungsfensters bei Beenden des Programms

Eigenschaftsknoten des Schließungsfensters

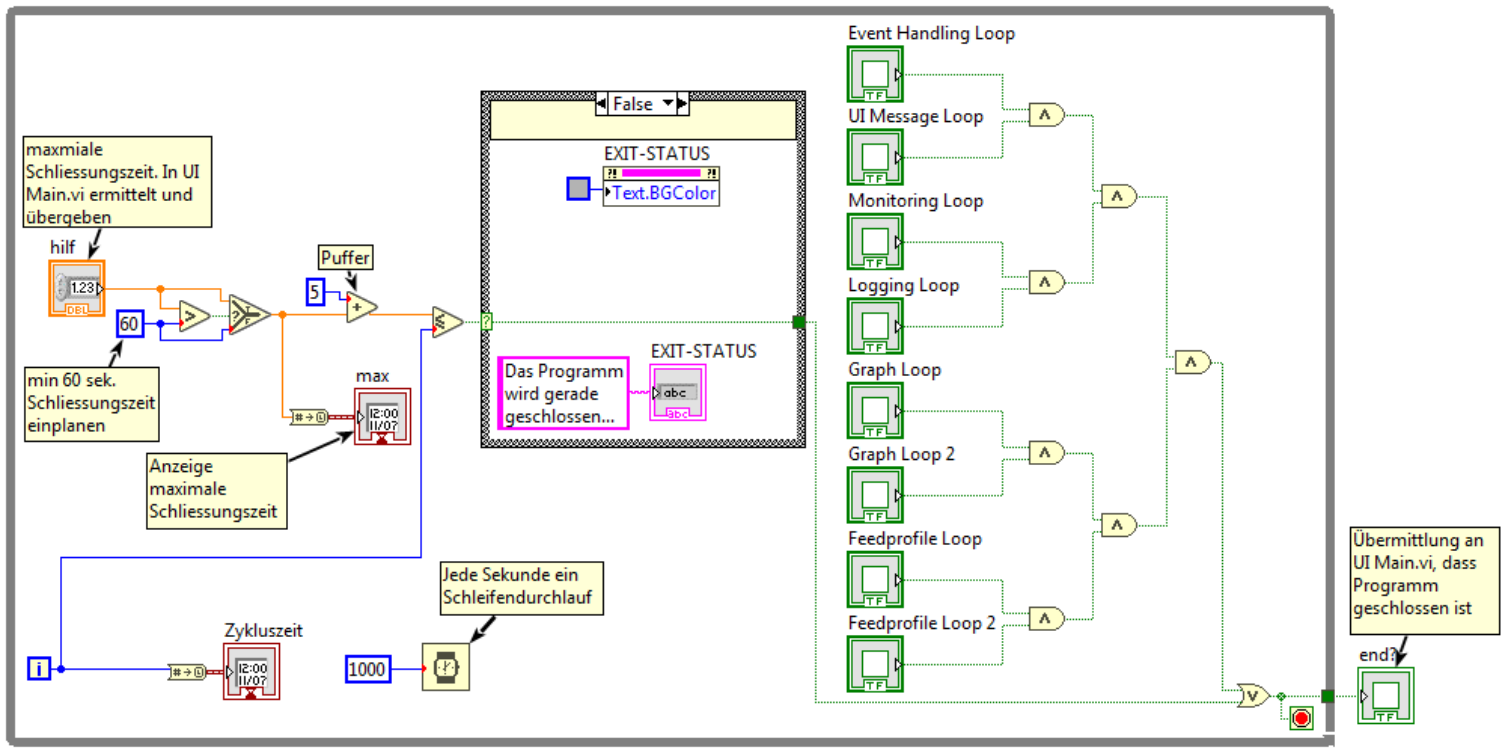


Abbildung: Ausgelagertes Schliessungsfenster in einem VI

Eigenschaftsknoten des Schließungsfensters

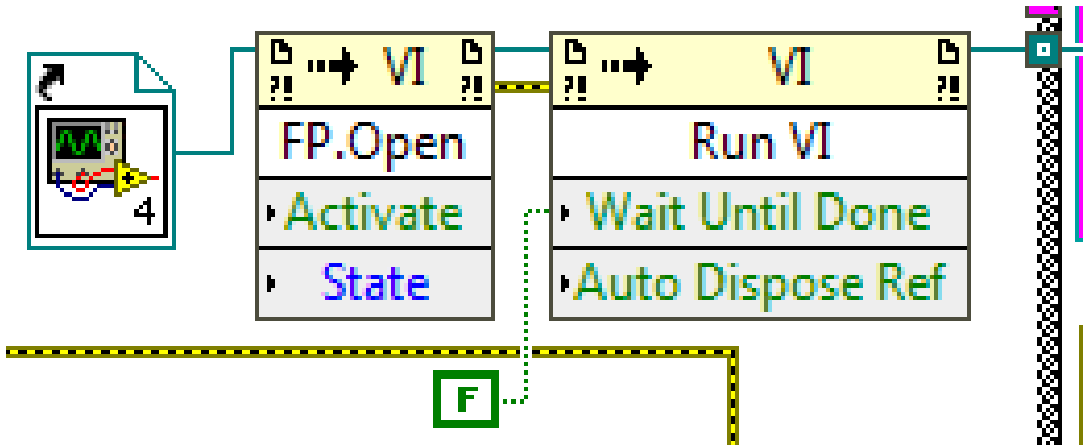


Abbildung: Aufruf des Schliessungsfenster-VI

Eigenschaftsknoten des Schließungsfensters

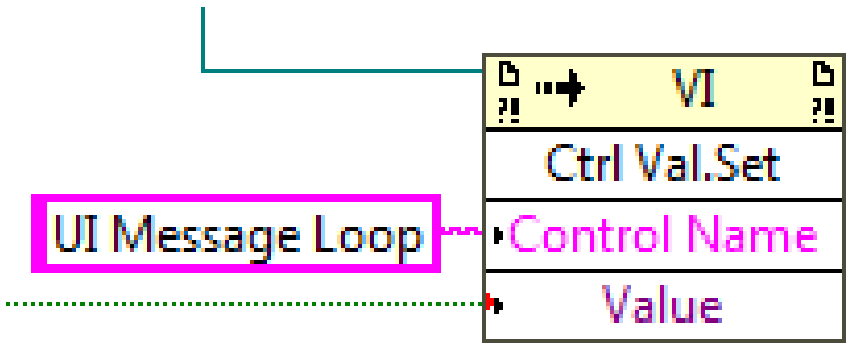


Abbildung: Zugriff auf die Frontpanelobjekte des SubVIs mit Methodenknoten

Eigenschaftsknoten des Schließungsfensters

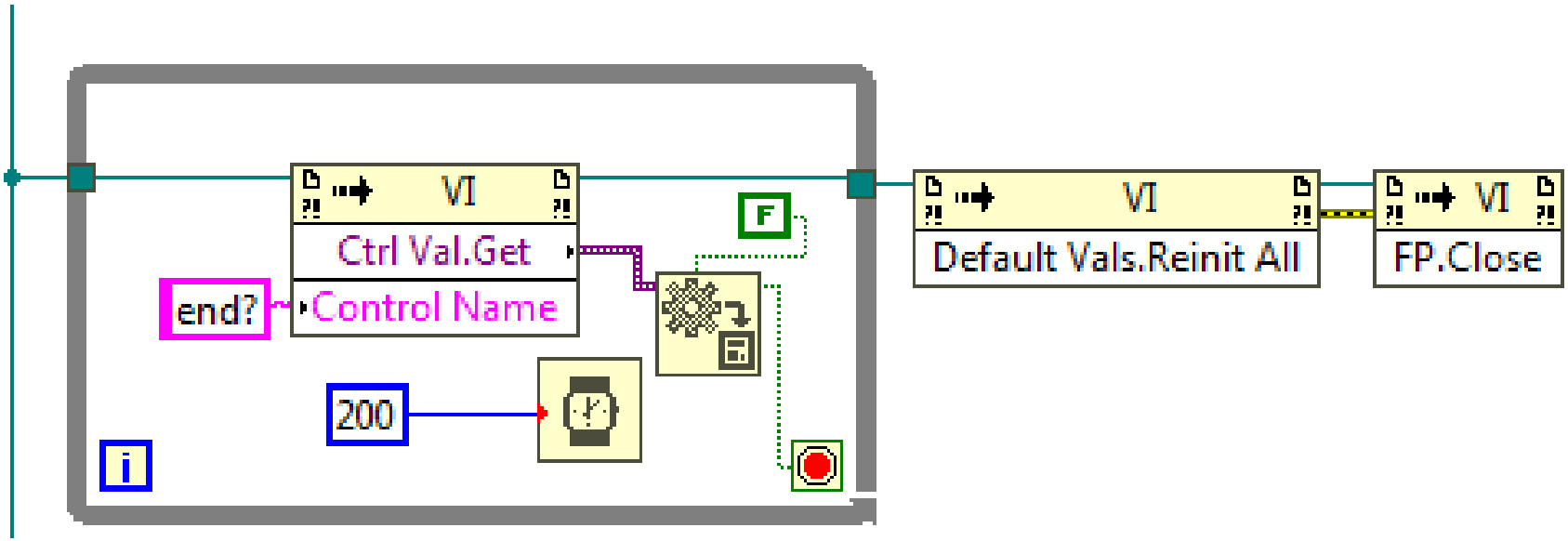


Abbildung: Beenden des Schliessungsfenster-VI

Eigenschaftsknoten des Schließungsfensters

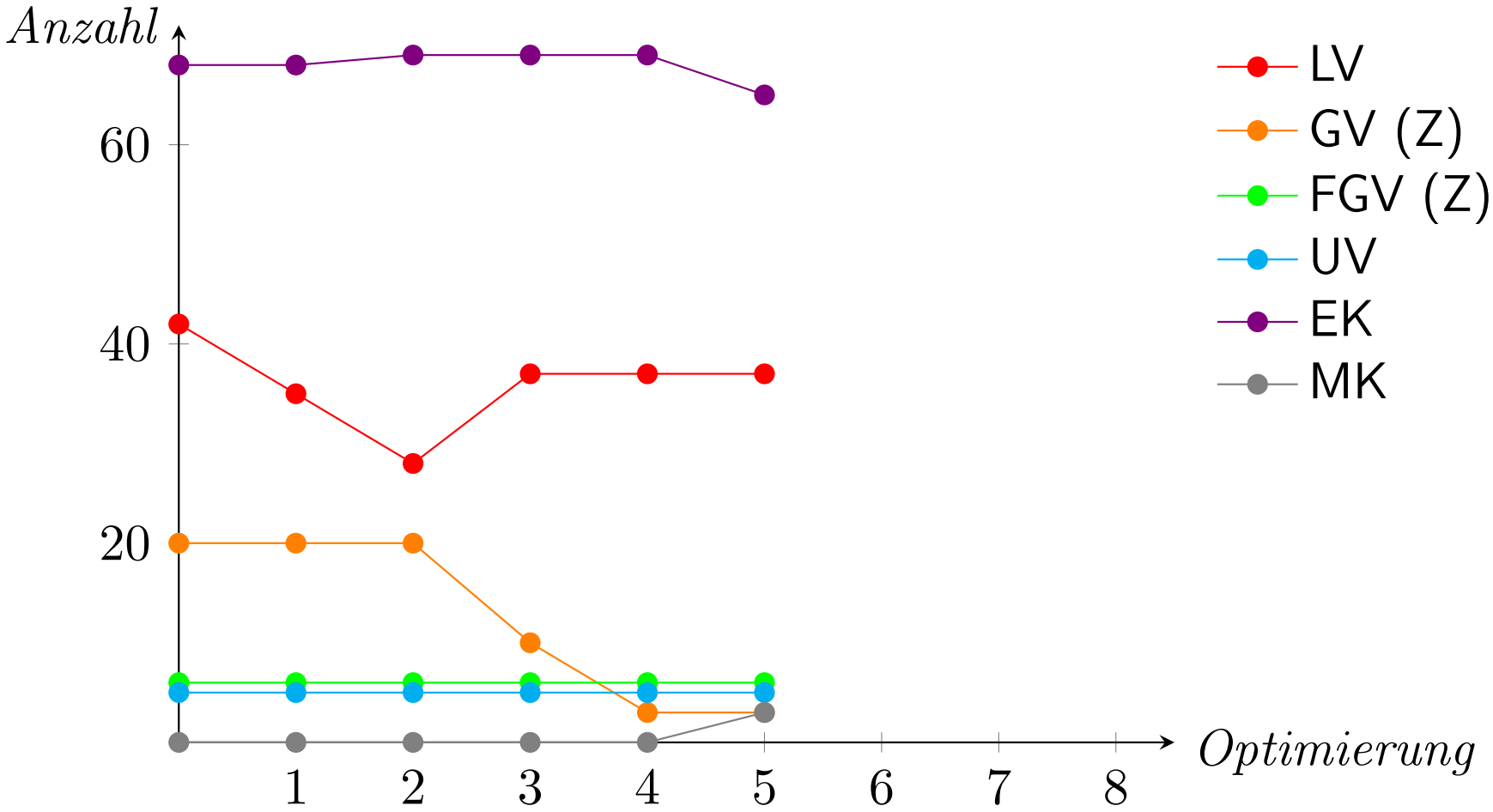


Abbildung: Darstellung des Optimierungsprozesses–Schritt 5

Eigenschaftsknoten des Schließungsfensters

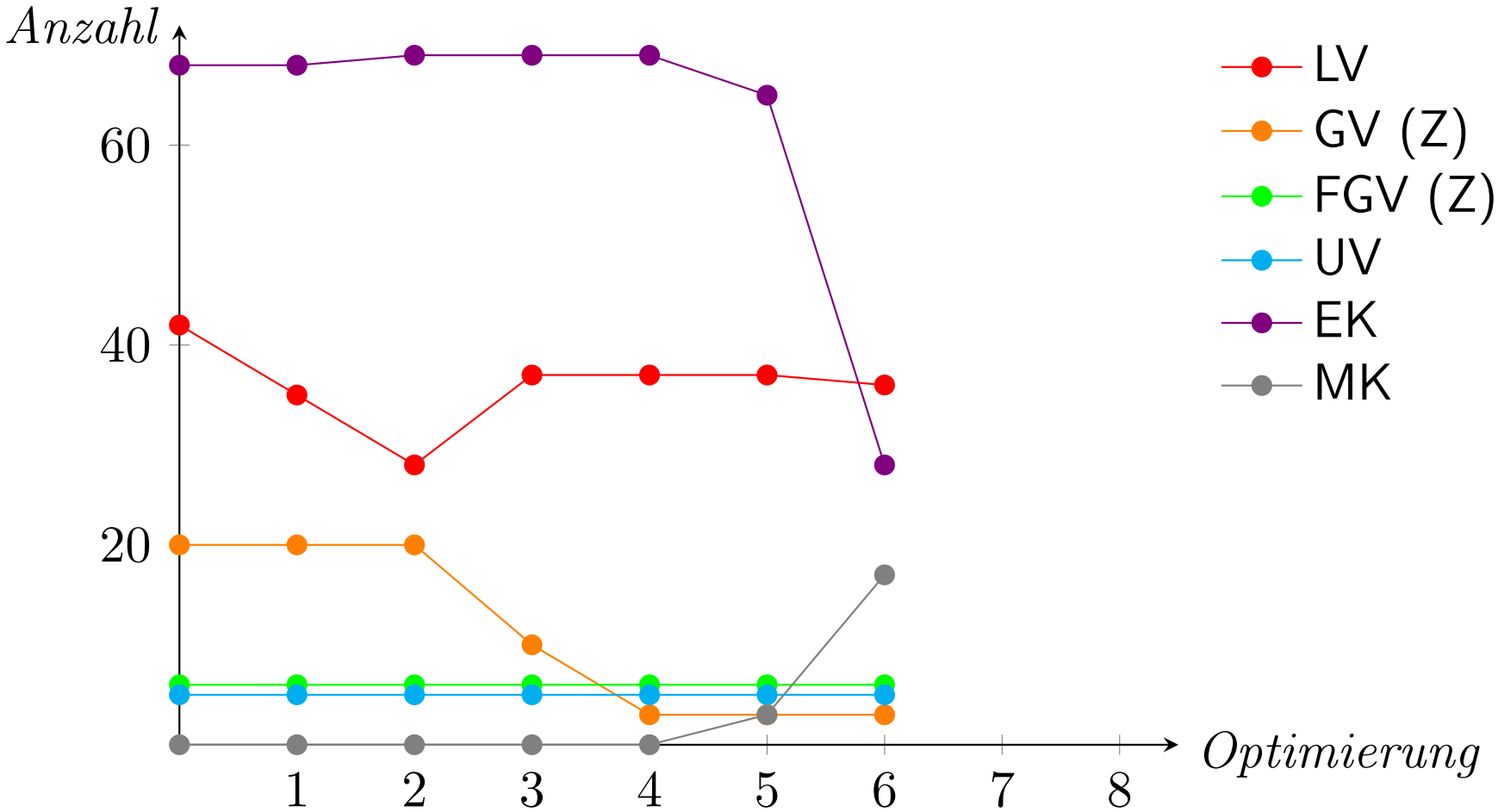


Abbildung: Darstellung des Optimierungsprozesses–Schritt 6

Umgebungsvariablen

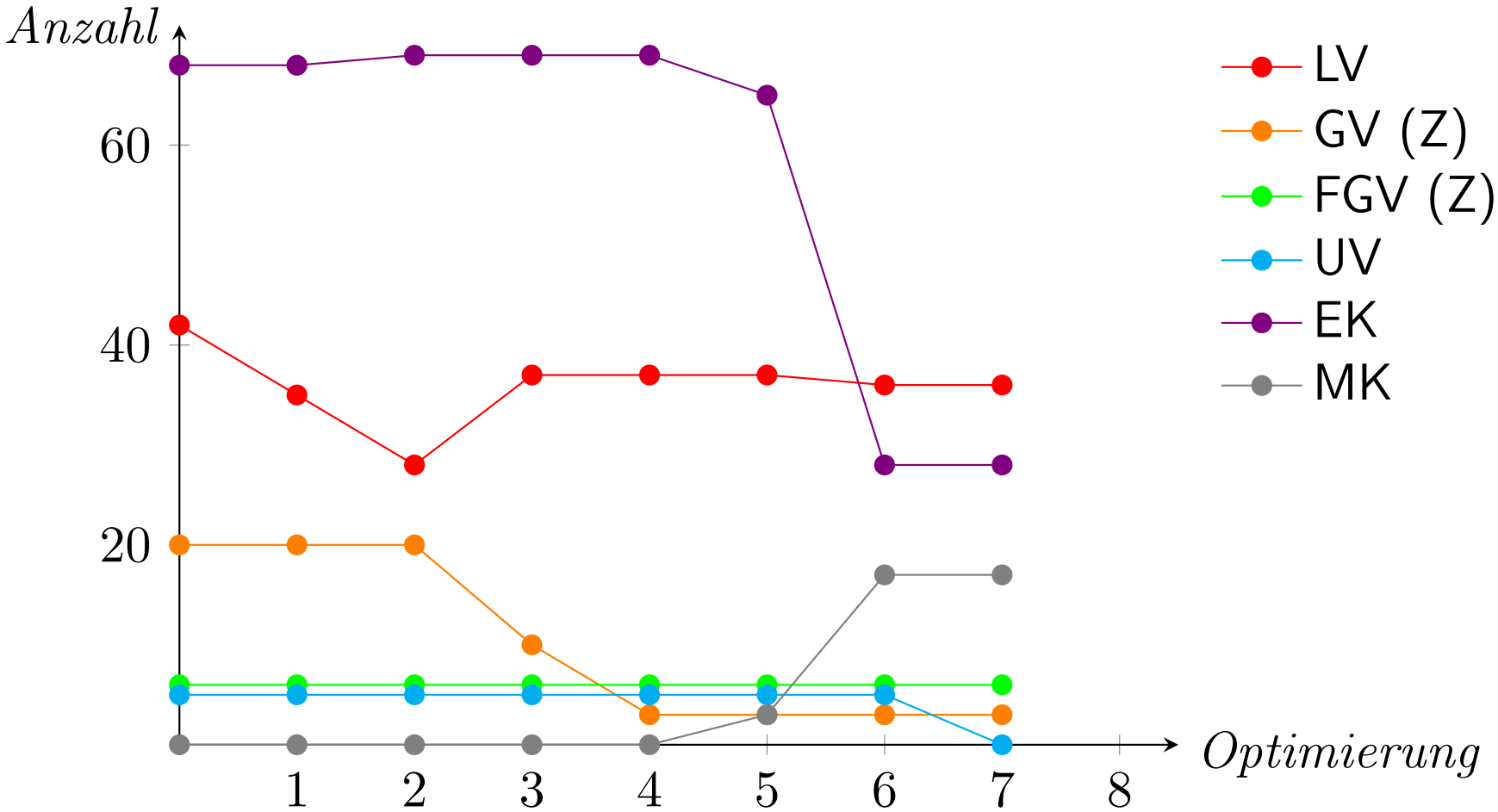


Abbildung: Darstellung des Optimierungsprozesses–Schritt 7

Weitere Optimierungen

> Queue ist zur internen Kommunikation sinnvoll

Weitere Optimierungen

> Queue ist zur internen Kommunikation sinnvoll

> Zugriffe auf funktionale globale Variable sinnvoll

Weitere Optimierungen

- Queue ist zur internen Kommunikation sinnvoll
- Zugriffe auf funktionale globale Variable sinnvoll
- - 1 Eigenschaftsknoten

Weitere Optimierungen

> Queue ist zur internen Kommunikation sinnvoll

> Zugriffe auf funktionale globale Variable sinnvoll

> - 1 Eigenschaftsknoten

> - 2 Lokale Variablen

Weitere Optimierungen

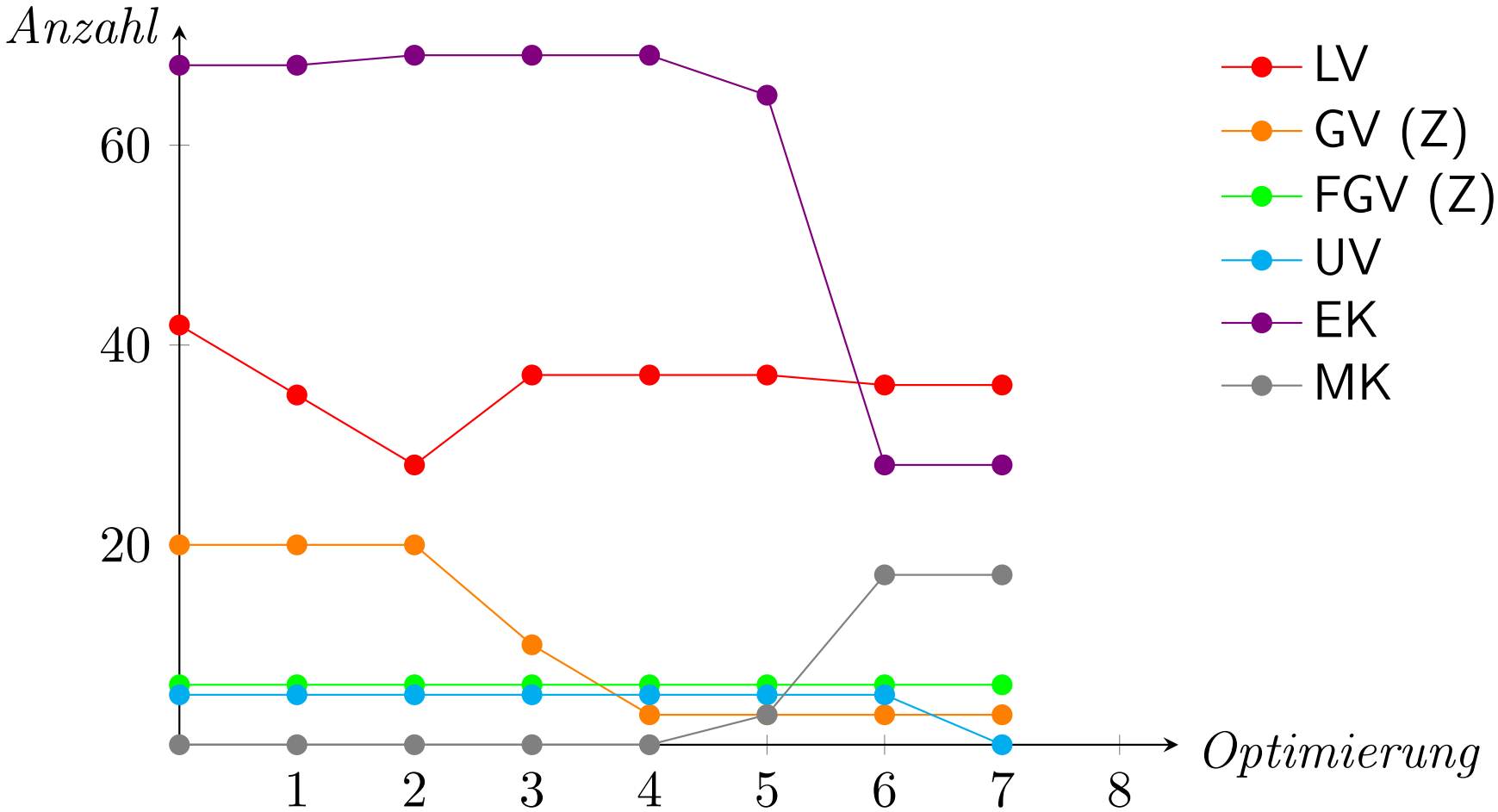


Abbildung: Darstellung des Optimierungsprozesses–Schritt 7

Weitere Optimierungen

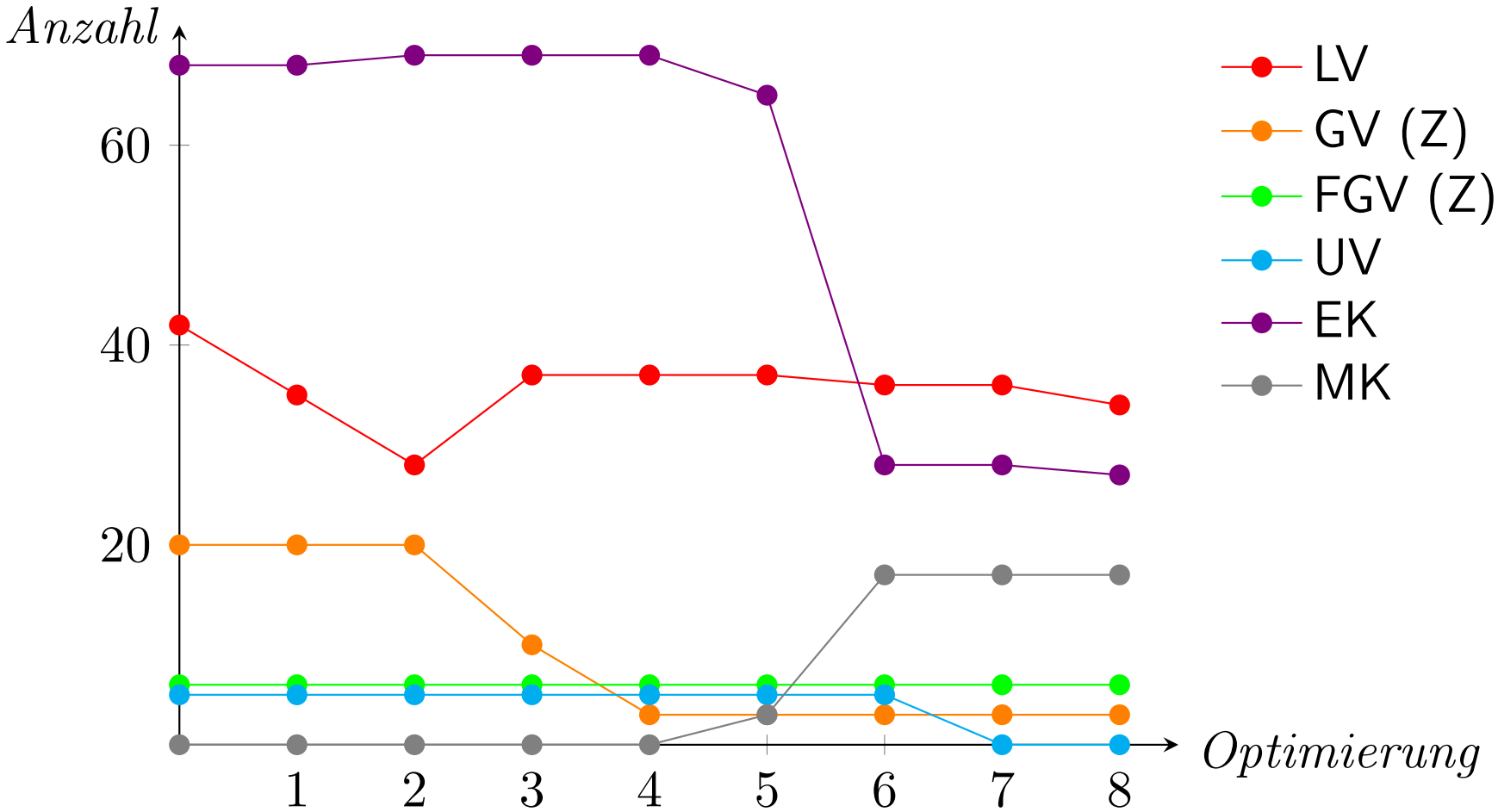


Abbildung: Darstellung des Optimierungsprozesses–Schritt 8

Ergebnis

Tabelle: Verwendung der verschiedenen Datenübertragungstypen im Glucose-Puls-Reaktor-Programm – vorher

Typ	B	B-T	T	T-F	F	T-R	F-R	B-R
Lokale	42	0	0	0	0	0	0	0
Globale (Z)	20	3	22	0	0	0	0	0
Funk. globale (Z)	6	0	0	0	0	0	0	0
Umgebungs. (Z)	5	12	0	0	0	6	0	40
E.Knoten	68	0	3	0	0	0	0	0
M.Knoten	0	0	0	1	0	0	0	0
R./W. Control	0	0	0	4	0	0	0	0
FIFO	0	0	0	1	0	0	0	0
Queue	1	0	1	0	0	0	0	0
Stream	0	2	0	0	0	0	0	0
FPGA I/O Node	0	0	0	0	0	0	2	0

Ergebnis

Tabelle: Verwendung der verschiedenen Datenübertragungstypen im Glucose-Puls-Reaktor-Programm – nachher

Typ	B	B-T	T	T-F	F	T-R	F-R	B-R
Lokale	34	0	0	0	0	0	0	0
Globale (Z)	3	3	22	0	0	0	0	0
Funk. globale (Z)	6	0	0	0	0	0	0	0
Umgebungs. (Z)	0	12	0	0	0	6	0	31
E.Knoten	27	0	3	0	0	0	0	0
M.Knoten	17	0	0	1	0	0	0	0
R./W. Control	0	0	0	4	0	0	0	0
FIFO	0	0	0	1	0	0	0	0
Queue	1	0	1	0	0	0	0	0
Stream	0	2	0	0	0	0	0	0
FPGA I/O Node	0	0	0	0	0	0	2	0

Ergebnis

Tabelle: Prozentuale Reduktion in Hinblick auf Datenübertragungstypen

Typ	Vorher	Nachher	Reduktion
Lokale Variablen	42	34	19%
Globale Variablen	20	3	85%
Funktionale globale Variablen	6	6	0%
Umgebungsvariablen	5	0	100%
Eigenschaftsknoten	68	27	60%
gesamt	141	70	50%

Ergebnis

Tabelle: Prozentuale Reduktion in Hinblick auf Datenübertragungstypen mit Berücksichtigung der Methodenknoten

Typ	Vorher	Nachher	Reduktion
gesamt ohne M.	141	70	50%
Methodenknoten	0	17	—
gesamt	141	87	48%

Ergebnis

Tabelle: Prozentuale Reduktion in Hinblick auf Datenübertragungstypen im gesamten Programm

Typ	Vorher	Nachher	Reduktion
Lokal	42	34	19%
Global	45	28	38%
Funk. global	6	6	0%
Umgebungs.	63	49	22%
E.Knoten	71	30	58%
M.Knoten	1	18	—
Andere	11	11	0%
gesamt	238	176	26%

Ausblick

Ausblick

> Optimierung der Anzahl der VIs

Ausblick

- Optimierung der Anzahl der VIs
- Benutzeroberfläche

Ausblick

- Optimierung der Anzahl der VIs
- Benutzeroberfläche
  - Erweiterung der Funktionen

Ausblick

- Optimierung der Anzahl der VIs
- Benutzeroberfläche
  - Erweiterung der Funktionen
  - Benutzerfreundlichkeit

Ausblick

- Optimierung der Anzahl der VIs
- Benutzeroberfläche
  - Erweiterung der Funktionen
  - Benutzerfreundlichkeit
- Motorsteuerung

Literatur I

Wolfgang Georgi und Ergun Metin. *Einführung in LabVIEW*. Deutsch. Bd. 3., vollständig überarbeitete und erweiterte Auflage. 2007. Kap. 1, S. 19–20. 454 S. ISBN: 978-3-446-41109-8.

National Instruments. *Globale Variablen*. Deutsch. Artikelnummer:371361H-0113; Besucht: 27.09.2017. URL: `http://zone.ni.com/reference/de-XX/help/371361H-0113/lvconcepts/glob_variables/`.

National Instruments. *Methoden des Datenaustauschs in LabVIEW*. Deutsch. Artikelnummer:371361L-0113; Besucht: 04.12.2017. URL: `http://zone.ni.com/reference/de-XX/help/371361L-0113/lvconcepts/data_comm/`.

Literatur II

National Instruments. *Streamen von Daten und Senden von Befehlen zwischen Applikationen*. Deutsch. Artikelnummer:371361H-0113; Besucht: 04.12.2017. URL: http://zone.ni.com/reference/de-XX/help/371361H-0113/lvconcepts/networkstreams/.

National Instruments. *Transferring Data between Devices or Structures Using FIFOs (FPGA Module)*. Englisch. Artikelnummer:371599H-01; Besucht:06.10.2017. URL: https://zone.ni.com/reference/en-XX/help/371599H-01/lvfpgaconcepts/fpga_transfer_data/.

Literatur III

National Instruments. *Umsichtige Verwendung lokaler und globaler Variablen*. Deutsch. Artikelnummer:371361H-0113; Besucht: 27.09.2017. URL: http://zone.ni.com/reference/de-XX/help/371361H-0113/lvconcepts/using_local_and_global/.

National Instruments. *Verwenden der Umgebungsvariablen in LabVIEW*. Deutsch. Besucht:05.10.2017. URL: http://www.ni.com/white-paper/4679/de/.

Literatur IV

National Instruments. *Vorschläge für die Verwendung von Ausführungssystemen und Prioritäten. Funktionale globale Variablen*. Deutsch. Artikelnummer:371361J-0113;Besucht:05.10.2017. URL: http://zone.ni.com/reference/de-XX/help/371361H-0113/lvconcepts/suggestions_for_exec/#Functional_Global_Variables.

Rahman Jamal und Andre Hagestedt. *LabVIEW Das Grundlagenbuch*. Deutsch. Bd. 4. Auflage. 2004. Kap. 13-Konventionelle Techniken in LabVIEW, S. 447–469. 560 S. ISBN: 3-8273-2051-8.

User. *Field Programmable Gate Array*. Deutsch. Besucht:04.10.2017. URL: https://de.wikipedia.org/wiki/Field_Programmable_Gate_Array.