Aufgabenblatt 4 ErDisSys

Aufgabe 1

a) – Grafische Erweiterung von Simulink zur Modellierung und Simulation von

   a. Ereignisgesteuerte reaktiven Systemen mit endlich vielen Zuständen

   b. Zustandsautomaten (engl.: Finite State Machines)

- Stateflow ist eine Software-Umgebung von MathWorks, die speziell für die Modellierung, Simulation und Implementierung von Zustandsmaschinen entwickelt wurde. Zustandsmaschinen sind ein Konzept aus der Informatik und der Systemtheorie, das verwendet wird, um das Verhalten eines Systems oder einer Komponente zu modellieren, das sich in verschiedenen Zuständen befinden kann.
- Mit Stateflow können Entwickler Zustandsdiagramme erstellen, die den Zustandsübergang und das Verhalten eines Systems beschreiben. Zustandsdiagramme bestehen aus Zuständen, Übergängen zwischen den Zuständen und Aktionen, die beim Übergang zwischen den Zuständen ausgeführt werden. Die Zustände repräsentieren die verschiedenen Phasen oder Modi, in denen sich ein System befinden kann, und die Übergänge zeigen die Bedingungen an, unter denen das System von einem Zustand in einen anderen wechselt. Die Aktionen definieren die Operationen oder Reaktionen, die beim Zustandsübergang ausgeführt werden sollen.
- Stateflow kann in Kombination mit MATLAB und Simulink verwendet werden, um das Verhalten von komplexen Systemen zu modellieren und zu simulieren. Es ermöglicht die Integration von Zustandsmaschinen in die allgemeine Systemmodellierung und -simulation und bietet eine visuelle Darstellung des Systemverhaltens, die leicht verständlich und überprüfbar ist.
- Stateflow wird in verschiedenen Anwendungsgebieten eingesetzt, einschließlich der Steuerungs- und Regelungstechnik, der Fahrzeugtechnik, der Robotik, der Luft- und Raumfahrt sowie der Kommunikationssysteme. Es bietet Entwicklern eine effiziente Methode zur Modellierung und Implementierung von komplexen Systemverhalten und hilft dabei, Fehler zu identifizieren, zu vermeiden und zu beheben.

b)

Chart:

Zustandsautomaten werden als Zustandsübergangsdiuagramme („Charts“) dargestellt

In Stateflow, einem Modellierungswerkzeug von MATLAB, ist ein "Chart" eine grafische Darstellung eines Zustandsautomaten. Ein Chart besteht aus verschiedenen Elementen wie Zuständen, Übergängen, Aktionen und Ereignissen, die es ermöglichen, das Verhalten eines Systems zu modellieren.

Ein Chart repräsentiert den Zustandsraum eines Systems und ermöglicht es, das Systemverhalten in Form von Zuständen und deren Übergängen zu beschreiben. Zustände repräsentieren die verschiedenen Zustände, in denen sich das System befinden kann,

während Übergänge die Bedingungen und Ereignisse darstellen, die den Wechsel von einem Zustand in einen anderen auslösen.

Ein Chart kann auch hierarchisch organisiert sein, wobei ein übergeordneter Chart mehrere Untercharts enthalten kann. Diese Hierarchie ermöglicht es, komplexe Systeme in kleinere und besser handhabbare Teile zu zerlegen.

In einem Chart können auch Aktionen definiert werden, die beim Eintritt oder Verlassen eines Zustands ausgeführt werden. Diese Aktionen können beispielsweise Berechnungen, Signalmanipulationen oder das Auslösen von Ereignissen umfassen.

Durch die Verwendung von Charts in Stateflow können komplexe Systeme auf intuitive Weise modelliert und verstanden werden. Charts bieten eine visuelle Darstellung des Systemverhaltens und ermöglichen es Entwicklern, den Zustandsfluss und die Interaktionen zwischen den Zuständen effizient zu entwerfen und zu analysieren.

State:

In Stateflow, einem Modellierungswerkzeug von MATLAB, repräsentiert ein "State" (Zustand) einen bestimmten Zustand oder eine spezifische Aktivität eines Systems. Ein Zustand ist definiert durch eine eindeutige Kombination von Eigenschaften, die das Verhalten und die Merkmale des Systems zu einem bestimmten Zeitpunkt beschreiben.

- Ein Zustand kann als eine Art Container betrachtet werden, der Aktionen, Übergänge und interne Variablen enthält. Zustände dienen dazu, das Verhalten eines Systems in unterschiedlichen Situationen oder Bedingungen zu beschreiben.

- Jeder Zustand kann eine oder mehrere der folgenden Eigenschaften haben:

- Aktionsblöcke: Diese Blöcke enthalten den auszuführenden Code oder die Aktionen, wenn der Zustand aktiv ist. Aktionen können Berechnungen, Signalmanipulationen, Funktionsaufrufe oder andere spezifische Operationen umfassen.

- Übergänge: Ein Zustand kann Übergänge zu anderen Zuständen haben, die den Wechsel von einem Zustand in einen anderen unter bestimmten Bedingungen auslösen. Diese Bedingungen können auf Ereignissen, Bedingungen oder zeitlichen Auslösern basieren.

- Eingangsereignisse: Zustände können Ereignisse empfangen, die von externen Quellen oder anderen Zuständen generiert werden. Diese Ereignisse können den Zustandsübergang oder das Verhalten im Zustand beeinflussen.

- Aktive Dauer: Ein Zustand kann eine bestimmte aktive Dauer haben, die angibt, wie lange der Zustand aktiv bleibt, bevor ein Übergang zu einem anderen Zustand ausgelöst wird.

- Durch die Kombination von Zuständen und Übergängen können komplexe Verhaltensmuster in einem Stateflow-Chart modelliert werden. Zustände ermöglichen es, das Verhalten eines Systems in einer strukturierten und intuitiven Weise darzustellen und zu organisieren.

History Junction:

Ein "History Junction" ist ein Konzept in Stateflow, einem Modellierungswerkzeug von MATLAB. Ein History Junction ermöglicht es, den Zustand eines übergeordneten Zustands zu speichern und beim erneuten Eintritt in den übergeordneten Zustand an der Stelle fortzufahren, an der er zuvor verlassen wurde. Es stellt eine Möglichkeit dar, den vorherigen Zustand wiederherzustellen, anstatt immer wieder von Anfang an zu beginnen.

Normalerweise erfolgt der Zustandsübergang in Stateflow von einem Zustand zum nächsten gemäß definierten Bedingungen. Beim Verlassen eines Zustands gehen alle internen Zustände und deren Zustandsverlauf verloren. In manchen Fällen kann es jedoch wünschenswert sein, den vorherigen Zustand fortzusetzen, wenn ein übergeordneter Zustand erneut betreten wird.

Hier kommt das Konzept des History Junctions ins Spiel. Ein History Junction ist eine spezielle Art von Übergang in einem Zustandsautomaten. Wenn ein Zustand einen History Junction-Übergang besitzt und der übergeordnete Zustand erneut betreten wird, überprüft Stateflow, ob ein vorheriger Zustand gespeichert wurde. Wenn ja, wird der Zustand an dieser Stelle fortgesetzt, anstatt vom Anfang zu beginnen. Dadurch wird der vorherige Zustand wiederhergestellt.

Die Verwendung eines History Junctions ermöglicht es, den Zustandsverlauf beizubehalten und das Verhalten eines Systems an der Stelle wieder aufzunehmen, an der es zuvor unterbrochen wurde. Dies kann insbesondere dann nützlich sein, wenn der Zustandsverlauf oder der Systemkontext wichtig sind und nicht bei jedem Eintritt in den übergeordneten Zustand verloren gehen sollen.

Default Transition:

Ein "Default Transition" ist eine Art Übergang in einem Zustandsautomaten, der ausgelöst wird, wenn keine anderen Übergänge die spezifischen Bedingungen erfüllen. Der Default Transition wird genutzt, um einen Standardpfad zu definieren, der genommen wird, wenn keine anderen Übergänge aktiviert werden können.

In einem Stateflow-Chart kann ein Zustand mehrere ausgehende Übergänge haben, die durch Bedingungen oder Ereignisse ausgelöst werden. Wenn keiner dieser Übergänge aktiviert wird, wird der Default Transition aktiviert und der Zustand wechselt entsprechend zu einem anderen Zustand.

Der Default Transition ist optional und muss nicht in jedem Zustand definiert werden. Wenn jedoch ein Default Transition in einem Zustand vorhanden ist und keine anderen Übergänge aktiviert werden, erfolgt der Zustandsübergang entsprechend dem Default Transition.

Der Default Transition kann als eine Art "Fallback" betrachtet werden, um sicherzustellen, dass der Zustand des Systems in einen anderen Zustand wechselt, auch wenn keine spezifischen Bedingungen erfüllt sind. Es bietet eine Möglichkeit, das Verhalten des Systems zu steuern, wenn keine anderen Übergänge aktiviert werden können.

Bei der Modellierung mit Stateflow ist es wichtig, die Verwendung von Default Transitions sorgfältig zu planen und zu überprüfen, um sicherzustellen, dass das Verhalten des Systems den Anforderungen und Spezifikationen entspricht.

Connective Junction:

Ein "Connective Junction" ist ein Konzept in Stateflow, einem Modellierungswerkzeug von MATLAB. Eine Connective Junction ermöglicht die Verbindung von mehreren ausgehenden Übergängen aus einem Zustand zu einem einzelnen Zielzustand.

Normalerweise können in einem Zustand mehrere ausgehende Übergänge zu verschiedenen Zielzuständen definiert werden. Diese Übergänge können durch Bedingungen oder Ereignisse ausgelöst werden. Bei Verwendung einer Connective Junction können diese ausgehenden Übergänge an einer einzigen Verbindungspunkt zusammengeführt werden, bevor sie den Zielzustand erreichen.

Die Connective Junction dient als Zwischenschritt, um mehrere Übergänge zu gruppieren und den Übergangsausgangspunkt in einem Zustand zu konsolidieren. Sie ermöglicht es, die Ausführung von Aktionen oder die Bedingungsüberprüfung an einem einzigen Ort zu zentralisieren, bevor der endgültige Zustandswechsel stattfindet.

Der Hauptvorteil der Verwendung einer Connective Junction besteht darin, dass der Zustandsautomat übersichtlicher und leichter zu verstehen ist. Anstatt mehrere separate Übergänge aus einem Zustand zu haben, die zu verschiedenen Zielzuständen führen, können diese Übergänge mit einer Connective Junction zusammengefasst werden, was die Lesbarkeit und Wartbarkeit des Stateflow-Modells verbessert.

Connective Junctions können insbesondere dann nützlich sein, wenn mehrere Übergänge aus einem Zustand ähnliche Bedingungen oder Aktionen aufweisen, die vor dem Zustandswechsel ausgeführt werden sollen. Durch die Verwendung einer Connective Junction können diese gemeinsamen Aktionen oder Bedingungen an einer einzigen Stelle definiert werden, was die Modellierungseffizienz erhöht.

Truth Table:

Kompakte Realisierung von logischen Wahrheitstabellen

- Vermeidung von unübersichtlichen UND/ODER-Verknüpfungen
- Werden wie grafische Funktionen verwendet
- Bestehen aus
- Funktionsprototyp (wie bei graph.Fnkt.)
- Wahrheitstabelle mit
- Condition-Table
- Decision-Table

D.h. jede Spalte Di entspricht einer IF-Bedingung
◊UND-Verknüpfungen aller Bedingungen in einer Spalte

◊Die Aktion der ersten wahren Spalte (von links nach rechts) wird ausgeführt (und dann das Truth Table sofort verlassen)

```
if ((x < 1) & (x < 2) & (x < 3) & (x < 4) & (x < 5))
        y = 0;
elseif (~(x < 1) & (x < 2) & (x < 3) & (x < 4) & (x < 5))
        y = 1;
elseif (~(x < 1) & ~(x < 2) & (x < 3) & (x < 4) & (x < 5))
        y = 2;
elseif (~(x < 1) & ~(x < 2) & ~(x < 3) & (x < 4) & (x < 5))
        y = 3;
elseif (~(x < 1) & ~(x < 2) & ~(x < 3) & ~(x < 4) & (x < 5))
        y = 4;
else
        y = 5;
end
```

In einer Truth Table werden alle möglichen Kombinationen von Eingangsvariablen aufgelistet, und für jede Kombination wird der entsprechende Zustandsübergang angegeben. Die Eingangsvariablen können Boolesche Werte oder diskrete Werte annehmen, je nachdem, wie sie im Modell definiert sind.

Die Truth Table wird in Stateflow verwendet, um den Zustandsautomaten zu konfigurieren und die Zustandsübergänge zu spezifizieren. Sie dient als Referenz für die Ausführung des Modells, um den nächsten Zustand auf der Grundlage der aktuellen Eingangswerte zu bestimmen.

Durch die Verwendung einer Truth Table in Stateflow wird die Zustandslogik lesbarer und leichter zu überprüfen, da alle Zustandsübergänge klar definiert sind und keine Unsicherheit über den nächsten Zustand besteht

Funktion:

In Stateflow, einer graphischen Programmiersprache zur Modellierung von Zustandsautomaten und Ereignissen in Simulink, repräsentiert eine Funktion einen blockartigen Container, der eine Gruppe von Aktionen oder Berechnungen enthält. Eine Funktion in Stateflow wird verwendet, um komplexe Logik zu organisieren und wiederverwendbaren Code zu erstellen.

Eine Funktion in Stateflow kann eine beliebige Anzahl von Eingabe- und Ausgabeparametern haben. Sie kann Aktionen ausführen, Variablen deklarieren, Berechnungen durchführen, Bedingungen überprüfen, Zustände ändern und andere Funktionen aufrufen. Funktionen können in einem Zustandsdiagramm oder in anderen Funktionen verwendet werden.

Durch die Verwendung von Funktionen in Stateflow können komplexe Aufgaben aufgeteilt und modularisiert werden, was die Lesbarkeit, Wartbarkeit und Wiederverwendbarkeit des Codes verbessert. Funktionen können auch Parameter übergeben bekommen und Rückgabewerte liefern, um die Kommunikation und den Informationsaustausch zwischen verschiedenen Teilen des Modells zu ermöglichen.

Embedded MATLAB Function?

In Stateflow, einer graphischen Programmiersprache zur Modellierung von Zustandsautomaten und Ereignissen in Simulink, repräsentiert eine Funktion einen blockartigen Container, der eine Gruppe von Aktionen oder Berechnungen enthält. Eine Funktion in Stateflow wird verwendet, um komplexe Logik zu organisieren und wiederverwendbaren Code zu erstellen.

Eine Funktion in Stateflow kann eine beliebige Anzahl von Eingabe- und Ausgabeparametern haben. Sie kann Aktionen ausführen, Variablen deklarieren, Berechnungen durchführen, Bedingungen überprüfen, Zustände ändern und andere Funktionen aufrufen. Funktionen können in einem Zustandsdiagramm oder in anderen Funktionen verwendet werden.

Durch die Verwendung von Funktionen in Stateflow können komplexe Aufgaben aufgeteilt und modularisiert werden, was die Lesbarkeit, Wartbarkeit und Wiederverwendbarkeit des Codes verbessert. Funktionen können auch Parameter übergeben bekommen und Rückgabewerte liefern, um die Kommunikation und den Informationsaustausch zwischen verschiedenen Teilen des Modells zu ermöglichen.

Box:

Stateflow-Box: In Stateflow werden Zustände und Funktionen in rechteckigen Boxen dargestellt. Eine Zustandsbox repräsentiert einen Zustand im Zustandsdiagramm, während eine Funktionsbox eine Funktion innerhalb des Zustandsdiagramms darstellt. Diese Boxen enthalten Logik, Aktionen und Transitionen, die den Zustandsautomaten steuern.

Boxes sind grafische Einheiten zur visuellen Organisation eines Charts

zur Zusammenfassung von beliebigen grafischen Elementen

um Charts übersichtlicher zu machen

Kein Einfluss auf Funktion des Modells bis auf Aufführungsreihenfolge: - von oben nach unten - von links nach rechts (ähnlich wie bei parallelen Superstates)

c)

1. Öffne MATLAB: Starte das MATLAB-Programm, indem du es aus dem Startmenü oder der Programmliste deines Betriebssystems auswählst.
2. Öffne ein Simulink-Modell: Lade das gewünschte Simulink-Modell, in dem du Stateflow verwenden möchtest, oder erstelle ein neues Modell.
3. Öffne den Stateflow-Editor: Wenn du das Simulink-Modell geöffnet hast, siehst du eine Palette mit verschiedenen Simulink-Blöcken. Suche nach dem Stateflow-Block (meistens als "Stateflow Chart" bezeichnet) und ziehe ihn in den Modellbereich.
4. Doppelklicke auf den Stateflow-Block: Klicke mit der linken Maustaste doppelt auf den Stateflow-Block, um den Stateflow-Editor zu öffnen.
5. Der grafische Editor für Stateflow öffnet sich: Nachdem du den Stateflow-Block doppelt angeklickt hast, wird der grafische Editor für Stateflow geöffnet. Hier kannst du Zustandsdiagramme erstellen, Zustandsübergänge definieren und die Logik deines Stateflow-Modells gestalten.

d)

Der Model Explorer ist ein Werkzeug, das in der MATLAB/Simulink-Umgebung verfügbar ist. Es handelt sich um ein interaktives Fenster, das es ermöglicht, auf eine strukturierte und übersichtliche Weise auf verschiedene Elemente eines Simulink-Modells zuzugreifen und diese zu verwalten. Der Model Explorer bietet eine alternative Sicht auf das Modell und erleichtert die Navigation, Inspektion und Bearbeitung der Modellkomponenten.

Hier sind einige Hauptfunktionen des Model Explorers:

1. Hierarchische Darstellung: Der Model Explorer zeigt das Simulink-Modell in einer hierarchischen Struktur an. Es ermöglicht das Ausklappen und Zusammenklappen von Komponenten, um eine bessere Übersicht und Navigation zu ermöglichen.
2. Elementinspektion: Durch Auswahl eines bestimmten Modell- oder Subsystemelements im Model Explorer können detaillierte Informationen zu diesem Element angezeigt werden. Es bietet einen schnellen Zugriff auf die Parameter, Eigenschaften und Einstellungen des ausgewählten Elements.
3. Verknüpfungen und Referenzen: Der Model Explorer zeigt die Verknüpfungen und Referenzen zwischen verschiedenen Modellkomponenten an. Dadurch können Abhängigkeiten und Beziehungen zwischen den Elementen visualisiert und verwaltet werden.
4. Such- und Filterfunktionen: Der Model Explorer verfügt über Such- und Filterfunktionen, mit denen bestimmte Modellkomponenten basierend auf Namen, Typen oder anderen Kriterien gefunden werden können. Dies erleichtert die Suche nach bestimmten Elementen in großen und komplexen Modellen.

5. Modifikationen und Aktualisierungen: Der Model Explorer ermöglicht das Hinzufügen, Umbenennen, Löschen und Aktualisieren von Modellkomponenten. Dadurch können Modelländerungen schnell und einfach vorgenommen werden, ohne den grafischen Editor zu öffnen.

Der Model Explorer ist ein nützliches Werkzeug für die Modellverwaltung, -navigation und -analyse in MATLAB/Simulink. Er verbessert die Effizienz bei der Arbeit mit Modellen und erleichtert das Verständnis und die Bearbeitung von Modellstrukturen und -elementen

e)

In Stateflow können einem Chart oder State Daten bzw. Events über Eingänge zugewiesen werden. Hier sind die Schritte, um einem Chart oder State Daten bzw. Events zuzuweisen:

1. Öffne den Stateflow-Editor: Öffne den Stateflow-Editor, indem du doppelt auf den entsprechenden Chart-Block oder State-Block im Simulink-Modell klickst.
2. Definiere Eingänge: Im Stateflow-Editor kannst du Eingänge für den Chart oder State definieren. Klicke mit der rechten Maustaste auf den Chart-Block oder State-Block und wähle "Properties" oder "Eigenschaften" aus, um die Eigenschaften des Blocks anzuzeigen.
3. Weise Eingänge zu: In den Eigenschaften des Charts oder States kannst du Eingänge definieren. Klicke auf den Reiter "Inputs" oder "Eingänge" und füge die gewünschten Eingänge hinzu. Du kannst sie benennen und ihren Datentyp festlegen.
4. Verwende Eingänge in Transitionen oder Aktionen: Sobald die Eingänge definiert sind, können sie in Transitionen oder Aktionen des Charts oder States verwendet werden. In einer Transition kannst du eine Bedingung festlegen, die auf den Wert des Eingangs prüft, um den Zustandsübergang auszulösen. In Aktionen kannst du den Wert des Eingangs lesen und damit Berechnungen durchführen oder Zustandslogik steuern.
5. Aktualisiere das Simulink-Modell: Nachdem du die Eingänge zugewiesen und verwendet hast, speichere den Stateflow-Editor und aktualisiere das Simulink-Modell. Dadurch werden die Änderungen übernommen und das Modell ist bereit, Daten bzw. Events über die definierten Eingänge zu empfangen und zu verarbeiten.
6. Durch die Zuweisung von Eingängen können Charts und States in Stateflow mit Daten bzw. Events interagieren und entsprechend reagieren. Es ermöglicht die flexible Steuerung des Zustandsverhaltens basierend auf externen Ereignissen oder Datenänderungen.

f)

In Stateflow-Charts können Kommentare eingefügt werden, um den Code zu dokumentieren oder zusätzliche Informationen bereitzustellen. Hier sind die Schritte, um Kommentare in einem Chart einzufügen:

1. Öffne den Stateflow-Editor: Öffne den Stateflow-Editor, indem du doppelt auf den entsprechenden Chart-Block im Simulink-Modell klickst.
2. Wähle das Tool "Comment" aus: Im Stateflow-Editor findest du in der Symbolleiste verschiedene Tools und Symbole. Klicke auf das "Comment"-Symbol, das oft wie ein

Sprechblasensymbol aussieht. Alternativ kannst du die Tastenkombination "Ctrl+Alt+C" verwenden.

3. Ziehe das Kommentar-Symbol in den Chart: Nachdem du das "Comment"-Tool ausgewählt hast, siehst du ein Kommentar-Symbol, das dem Mauszeiger folgt. Klicke an der Stelle im Chart, an der du den Kommentar platzieren möchtest, und ziehe das Symbol, um die Größe des Kommentars festzulegen.
4. Schreibe den Kommentar: Sobald das Kommentar-Symbol platziert ist, wird ein Textfeld geöffnet, in dem du den Kommentar eingeben kannst. Schreibe den gewünschten Text und bestätige ihn mit der Eingabetaste.
5. Formatieren des Kommentars (optional): Du kannst den Kommentar auch formatieren, indem du beispielsweise die Schriftgröße oder den Textstil änderst. Dazu kannst du das Textfeld des Kommentars auswählen und auf die entsprechenden Formatierungsoptionen in der Symbolleiste des Stateflow-Editors zugreifen.
6. Positionieren und verschieben des Kommentars: Du kannst den Kommentar im Chart beliebig positionieren und verschieben, indem du das Kommentar-Symbol mit der Maus ziehst. Dadurch kannst du den Kommentar an einer geeigneten Stelle im Chart platzieren.

g)

In Stateflow haben die Begriffe "Exklusiv" und "Parallel" verschiedene Bedeutungen, je nachdem, in welchem Zusammenhang sie verwendet werden. Hier sind die beiden möglichen Bedeutungen:

Exklusiv (Exclusive):

1. Exklusive Verbindung: Im Kontext von Zustandsübergängen bedeutet "exklusiv", dass nur ein einzelner Übergang zu einem bestimmten Zeitpunkt aktiv sein kann. Wenn mehrere Übergänge gleichzeitig aktiviert werden könnten, wird der Übergang mit der höchsten Priorität oder den spezifischsten Bedingungen ausgeführt.

   Exklusiver Zustand: Ein exklusiver Zustand bezeichnet einen Zustand, der einen bestimmten Satz von Aktivitäten oder Aktionen ausführt, solange er aktiv ist. Andere Zustände im gleichen Zustandsdiagramm werden in diesem Fall nicht aktiviert.

Parallel:

2. Parallele Aktivität: Im Kontext von Zuständen ermöglicht der parallele Verbindungstyp die gleichzeitige Aktivierung mehrerer Zustände. Dadurch können mehrere Aktivitäten oder Aktionen parallel ablaufen. Parallele Aktivitäten können entweder explizit als parallele Zustände im Zustandsdiagramm modelliert werden oder durch die Verwendung von Junctions oder inneren Transitionen erreicht werden.

   Paralleler Zustand: Ein paralleler Zustand bezieht sich auf einen Zustand, der parallel zu anderen Zuständen aktiv sein kann. Dadurch können mehrere parallele Aktivitäten innerhalb des Modells stattfinden. Parallele Zustände können in Stateflow-Modellen verwendet werden, um komplexes Verhalten zu modellieren, das gleichzeitig oder unabhängig voneinander ablaufen kann.

Es ist wichtig zu beachten, dass die genaue Bedeutung von "exklusiv" und "parallel" stark vom Kontext innerhalb des Stateflow-Modells abhängt. Die Verwendung dieser Konzepte hängt von den spezifischen Anforderungen und der Modelllogik ab, die du in deinem Stateflow-Modell implementieren möchtest.

h)

In Stateflow kann ein Chart nicht explizit auf den "Exklusiv"-Modus eingestellt werden. Der Exklusivitätsaspekt in Stateflow bezieht sich auf die Logik der Zustandsübergänge und Zustände im Chart.

Wenn du sicherstellen möchtest, dass nur ein Übergang zu einem bestimmten Zeitpunkt aktiv ist, kannst du dies durch die richtige Definition der Bedingungen und Prioritäten der Zustandsübergänge erreichen. Die Priorität eines Übergangs kann durch die Reihenfolge der Übergänge im Zustandstisch oder durch die Verwendung von Guard-Bedingungen gesteuert werden. Der Übergang mit der höchsten Priorität oder der spezifischsten Bedingung wird aktiviert, während die anderen Übergänge inaktiv bleiben.

Es ist wichtig, die Zustände und Übergänge in deinem Chart sorgfältig zu modellieren und sicherzustellen, dass die exklusive Logik deinen Anforderungen entspricht. Du kannst die Bedingungen und Prioritäten der Zustandsübergänge entsprechend anpassen, um sicherzustellen, dass nur der gewünschte Übergang zu einem bestimmten Zeitpunkt aktiv ist.

Es gibt keine spezielle Einstellung oder Schalter, um den Chart selbst auf "Exklusiv" zu setzen. Die Exklusivität wird durch die Definition der Zustandsübergänge und deren Bedingungen innerhalb des Charts erreicht.

i)

Das Label einer Transition in Stateflow beschreibt die Bedingung oder das Ereignis, das erfüllt sein muss, damit der Übergang aktiviert wird. Das Label gibt an, unter welchen Umständen der Zustandsübergang von einem Zustand zum anderen erfolgen soll.

Das Label einer Transition kann verschiedene Formen haben, abhängig von der Art des Übergangs:

1. Bedingtes Label: Eine Transition kann ein bedingtes Label haben, das eine logische Bedingung enthält. Diese Bedingung wird evaluiert, und wenn sie wahr ist, wird der Übergang aktiviert. Ein Beispiel für ein bedingtes Label könnte "condition > 0" sein, wobei "condition" eine Variable oder ein Ausdruck ist, der eine numerische Bedingung repräsentiert.

2. Event Label: Eine Transition kann auch ein Event Label haben, das das Auftreten eines bestimmten Ereignisses auslöst. Das Ereignis kann entweder von externen Quellen wie Sensoren oder anderen Komponenten des Systems ausgelöst werden oder innerhalb des Stateflow-Modells generiert werden. Ein Event Label kann beispielsweise "event_name" sein, wobei "event_name" der Name des Ereignisses ist.

3. Timer Label: Eine Transition kann auch ein Timer Label haben, das eine Zeitverzögerung angibt, bevor der Übergang aktiviert wird. Das Timer Label kann eine Zeitdauer in Sekunden oder eine Variable enthalten, die die Zeitverzögerung repräsentiert. Ein Timer Label könnte beispielsweise "timer > 5" sein, wobei "timer" eine Variable ist, die die Zeit in Sekunden zählt, und der Übergang nach einer Verzögerung von mehr als 5 Sekunden aktiviert wird.

Das Label einer Transition spielt eine entscheidende Rolle bei der Steuerung des Zustandsübergangsverhaltens in Stateflow. Es ermöglicht es, die Bedingungen oder Ereignisse zu definieren, unter denen der Übergang stattfinden soll, und beeinflusst somit den Fluss und die Logik des Modells.

j)

Um den Debugger in Stateflow zu starten, folge diesen Schritten:

1. Öffne den Stateflow-Editor: Öffne den Stateflow-Editor, indem du doppelt auf den entsprechenden Chart-Block im Simulink-Modell klickst.
2. Aktiviere den Debugging-Modus: Im Stateflow-Editor gibt es eine Symbolleiste mit verschiedenen Schaltflächen. Suche nach dem "Debugging Mode"-Symbol, das oft wie ein kleines grünes Insekt aussieht. Klicke auf diese Schaltfläche, um den Debugging-Modus zu aktivieren. Alternativ kannst du auch die Tastenkombination "Ctrl+D" verwenden.
3. Setze Breakpoints: Um den Ablauf des Modells an bestimmten Punkten anzuhalten und den Debugger zu aktivieren, setze Breakpoints an den gewünschten Stellen. Klicke dazu mit der rechten Maustaste auf den Zustand oder die Transition, an der du einen Breakpoint setzen möchtest, und wähle "Set Breakpoint" oder "Breakpoint setzen" aus dem Kontextmenü aus. Du kannst auch eine bestimmte Aktion in einer Aktionssprache oder einem MATLAB-Code mit einem Breakpoint markieren.
4. Starte die Simulation: Starte die Simulation deines Simulink-Modells. Dazu kannst du den Simulationsstart-Button in der Simulink-Oberfläche verwenden.
5. Überwache den Debugging-Ablauf: Sobald die Simulation läuft und ein Breakpoint erreicht wird, stoppt die Ausführung an diesem Punkt, und der Debugger tritt in den Vordergrund. Du kannst Variablenwerte überprüfen, den Ablauf Schritt für Schritt verfolgen, den Zustand des Modells überwachen und Debugging-Aktionen durchführen, wie z.B. das Setzen von Haltepunkten, das Überwachen von Variablen oder das Verändern des Modellzustands.