1 Example Protocol

To illustrate the modeling features of BUSpec, we consider a simple Bus protocol that supports only incrementing burst transfers of unspecified length (INCR). The Bus supports any interleaving of read and write transfer sequences. In addition, there may be pipelined transactions where the address phase of one transfer may overlap with the data phase of the previous. Moreover, the Bus supports multiple masters and in case of consecutive transfer sequences, Bus handover can take place at the last transfer of a burst/sequence. Any such back-to-back pipelined transfer is specified in terms of a set of transfer sequences. The transfer mode indicates the status of the transfer in the current cycle. The start transfer mode is designated as IDLE since this is the idle state of the Bus when no transfer is taking place. The following transfer is represented as FST and this stands for the actual start of the transfer. The intermediate transfers in the sequence are indicated by INTR, and there may be multiple executions of this till the final transfer which is designated by transfer mode END.

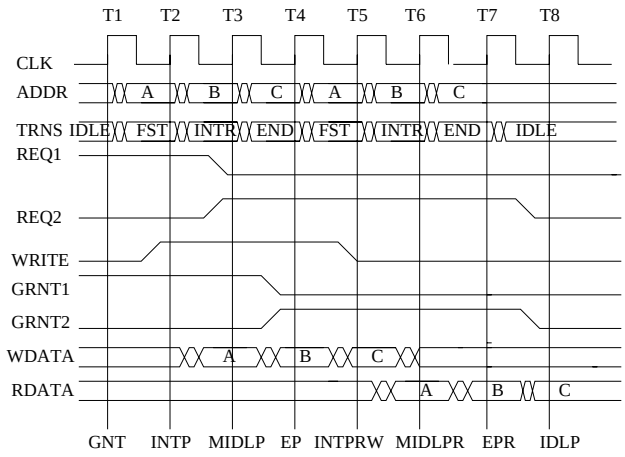

Figure 1: INCR back-to-back Transfer Sequence

Fig 1 shows the timing diagram for a 3-beat INCR back-to-back (write followed by read) transfer sequence (involving two different masters, master1 executing a write transfer and master2 executing a read transfer). An execution of the back-to-back transfer sequence consists of 8 basic phases as shown (A phase consists of the set of signal values and additional constraints at a time instant on the timing diagram). At the beginning of the protocol (GNT phase), master1 is granted access to the Bus. For a transfer to begin, the granted master must drive the address and the data for the first beat onto the Bus, which it does respectively in the phases INTP and MIDLP. In addition, in the phase MIDLP, the address of the next beat is also driven and the master lowers its request signal to signify that this is for the penultimate beat in the transfer sequence. In the subsequent phase EP, the data for the second beat and the address for the final beat are driven onto the Bus by master1 and the Bus is handed over to the other master which is granted. The next phase INTPRW shows an interesting phenomenon on the Bus since at this phase, master1 drives the last address for its write transfer and master2 also begins using the Bus by driving the transfer information for its read transfer. The rest of the phases correspond to the read transfer for master2 and finally the Bus moves to the idle phase in IDLP.

We now explain how to model the behavior as explained in Fig 1 using theBUSpec language.

1. **Specification of phases:** The transfer shown in Fig 1 has eight phases, namely *GNT, INTP, MIDLP, EP, INTRP, MIDLPR, EPR and IDLP*. In addition to signal values, the phase specification also contains some predefined predicates and functions. Table 1 shows three such predicates and functions along with their semantics. Table 2 shows the specification of the phase *EP*.

Predicate/Function	Remarks
Valid(x)	x is not in high-impedance or unknown state
Past(x)	Value of x at the previous clock cycle
Equal(x,y)	x has the value y in the current phase

Table 1: Predefined predicates and functions

2. **Specification of transfers:** A transfer is specified as a finite state machine consisting of the phases as states and some inter phase transitions. Table 3 shows the specification of the INCR write transfer in BUSpec. Fig 2 shows the state machine it represents.

StartPhase EP {
signal { **TRNS=‘END**, **REQ1**=0, **REQ2**=1, **GRNT1**=0, **GRNT2**=1, **WRITE**=1; }
Valid(**ADDR**), Valid(**WDATA**),
Equal(**ADDR**, past(**ADDR**) + size);
}
EndPhase

Table 2: Specification of the EP phase

T1 {	T2 {	T3 {	T4 {
INTP	MIDLP	MIDLP	INTP
MIDLP	MIDLP	EP	EP
}	}	}	}

Table 3: Inter phase transitions for INCR Write Transfer

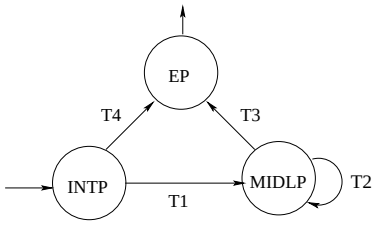

Figure 2: Phase-level state machine for INCR Write Transfer

3. **Specification of the system state machine:** The system level state machine consists of the set of transfers as states and inter transfer transitions connecting the states. *Inter-transfer* transitions are transitions between phases of two transfers. Fig 3 shows the detailed system level machine for our protocol. A rectangle in the figure represents a phase level state machine for a particular transfer and a transition between two such rectangles (e.g. IT1, IT2) represents an inter-transfer transitions. These transitions are also specified exactly like the inter-phase transitions. Note that some additional phases (e.g. INTPW, INTPR, INTPRD, RINTPRD) are introduced in the transfers 3. These phases are required for the complete protocol functionality. Phases INTPW and RINTPRD takes care of the write data and read data during consecutive write and read transfers. Phase INTPRD takes care of the read data during a back-to-back read-write transfer sequence. INTPR is a normal FST phase in a read transfer which doesn’t care about read or write data as no data transfer is pending by any previous pipelined transfer.

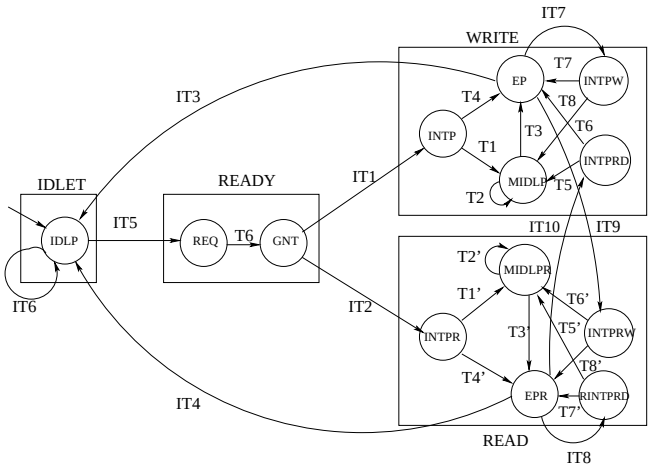

Figure 3: System-level state machine for INCR Transfer

The transfer level behavior of the protocol can be easily captured by an abstract state machine derived

from the system level machine as explained earlier. Fig 4 shows the transfer-level state machine for our protocol. The complete BUSpec specifcation of the above protocol is given below. Note that there is a

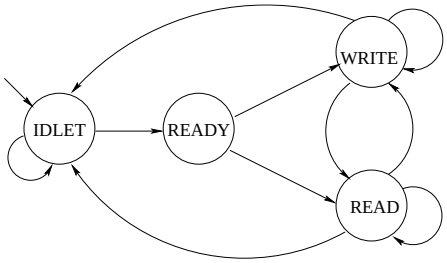

Figure 4: Transfer-level state machine for INCR Transfer Sequence

table for each of the supported transfers and inter-phase transitions are associated in the table. However inter-transfer transitions are modeled in a separate table. We have used the tables for readability. In practice you don't need these formats.

StartFSM
StartTransfer IDLE

StartPhase IDLP {
signal {
TRNS=‘IDLE, **GRNT1**=0, **GRNT2**=0,
REQ1=0, **REQ2**=0;
}
}
EndPhase

EndTransfer

Table 4: Specification of IDLE transfer

StartTransfer READY	

StartPhase REQ {
signal {
TRNS=‘IDLE;
}
ONE(**REQ**;
}
EndPhase
	StartPhase GRNT {
	signal {
	TRNS=‘IDLE;
	}
	ONE(**REQ**), ONE(**GRNT**);
	}
	EndPhase

StartPhTrans	

T6 {
REQ GRNT
}	

EndPhTrans	
EndTransfer	

Table 5: Specification of READY transfer

StartTransfer WRITE		

StartPhase INTP {
signal {
TRNS=‘FST, **WRITE**=1, **REQ1**=1,
REQ2=0, **GRNT1**=1 **GRNT2**=0;
}
Valid(**ADDR**);
}
EndPhase
	StartPhase MIDLP {
	signal {
	TRNS=‘INTR, **WRITE**=1, **REQ1**=0,
	REQ2=1, **GRNT1**=1 **GRNT2**=0;
	}
	Valid(**ADDR**), Valid(**WDATA**),
	Equal(**ADDR**, past(**ADDR**)+**size**);
	}
	EndPhase
		StartPhase EP {
		signal {
		TRNS=‘END, **WRITE**=1,
		REQ1=0,
		REQ2=1, **GRNT1**=0,
		GRNT2=0;
		}
		Valid(**ADDR**), Valid(**WDATA**),
		Equal(**ADDR**, past(**ADDR**) + **size**);
		}
		EndPhase

StartPhase INTPW {
signal {
WRITE=1, **REQ1**=0, **GRNT1**=0,
REQ2=1, **GRNT2**=1, **TRNS**=‘FST;
}
Valid(**ADDR**), Valid(**WDATA**);
}
EndPhase
	StartPhase INTPRD {
	signal {
	WRITE=1, **REQ1**=0, **GRNT1**=0,
	REQ2=1, **GRNT2**=1, **TRNS**=‘FST;
	}
	Valid(**ADDR**), Valid(**RDATA**);
	}
	EndPhase	

StartPhTrans		

T1 {
INTP MIDLP
}
	T2 {
	MIDLP MIDLP
	}
		T3 {
		MIDLP EP
		}

T4 {
INTP EP
}
	T5 {
	INTPW EP
	}
		T6 {
		INTPW MIDLP
		}

T7 {
INTPRD MIDLP
}
	T8 {
	INTPRD MIDLP
	}	

EndPhTrans		
EndTransfer		

Table 6: Specification of WRITE transfer

StartTransfer READ		

StartPhase INTPR {
signal {
TRNS=‘FST, **WRITE**=0, **REQ1**=1,
REQ2=0, **GRNT1**=1 **GRNT2**=0;
}
Valid(**ADDR**);
}
EndPhase
	StartPhase MIDLPR
	signal {
	TRNS=‘INTR, **WRITE**=0, **REQ1**=0,
	REQ2=1, **GRNT1**=1 **GRNT2**=0;
	}
	Valid(**ADDR**), Valid(**RDATA**),
	Equal(**ADDR**, past(**ADDR**)+**size**);
	}
	EndPhase
		StartPhase EPR {
		signal {
		TRNS=‘END, **WRITE**=0,
		REQ1=0,
		REQ2=1, **GRNT1**=0,
		GRNT2=0;
		}
		Valid(**ADDR**), Valid(**RDATA**),
		Equal(**ADDR**, past(**ADDR**) + **size**);
		}
		EndPhase

StartPhase RINTPRD {
signal {
WRITE=0, **REQ1**=0, **GRNT1**=0,
REQ2=1, **GRNT2**=1, **TRNS**=‘FST;
}
Valid(**ADDR**), Valid(**RDATA**);
}
EndPhase
	StartPhase INTPRW {
	signal {
	WRITE=0, **REQ1**=0, **GRNT1**=0,
	REQ2=1, **GRNT2**=1, **TRNS**=‘FST;
	}
	Valid(**ADDR**), Valid(**WDATA**);
	}
	EndPhase	

StartPhTrans		

T1’ {
INTPR MIDLPR
}
	T2’ {
	MIDLPR MIDLPR
	}
		T3’ {
		MIDLPR EPR
		}

T4’ {
INTPR EPR
}
	T5’ {
	INTPRW EPR
	}
		T6’ {
		INTPRW MIDLPR
		}

T7’ {
RINTPRD MIDLPR
}
	T8’ {
	RINTPRD MIDLPR
	}	

EndPhTrans		
EndTransfer		

Table 7: Specification of READ transfer

StartSmTrans				

IT1 {
GNT INTP
}
	IT2 {
	GNT INTPR
	}
		IT3 {
		EP IDLP
		}
			IT4 {
			EPR IDLP
			}
				IT5 {
				IDLP REQ
				}

IT6 {
IDLP IDLP
}
	IT7 {
	EP INTPW
	}
		IT8 {
		EPR RINTPRD
		}
			IT9 {
			EP INTPRW
			}
				IT10 {
				EPR INTPRD
				}

EndSmTrans				
EndFSM				

Table 8: Specification of *inter-transfer* transitions