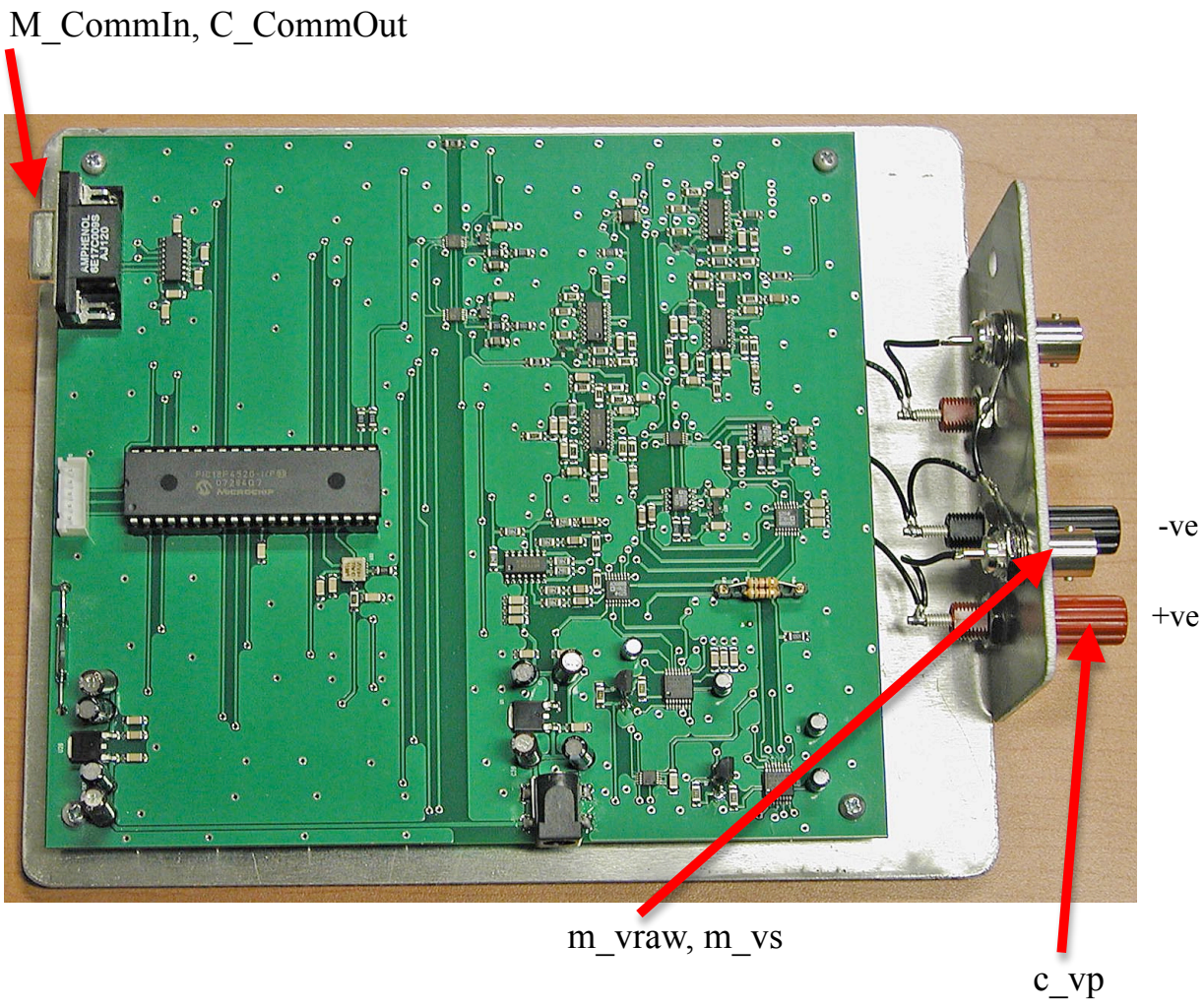
3K04 Requirements Specification of the Pacemaker

Author: Alan Wassyng

Revision: 2

Date: 27 September, 2011



1. Types

y_pacingMode = {Off, AAT, VVT, AOO, AAI, VOO, VVI, VDD, DOO, DDI, DDD, AOOR, AAIR, VOOR, VVIR, VDDR, DOOR, DDIR, DDDR}

y_pacingState = {PERMANENT, TEMPORARY, PACE_NOW, MAGNET, POWER_ON_RESET}

y_magnet = {INPLACE, NOT_INPLACE}

2. Variables – relevant to VVI

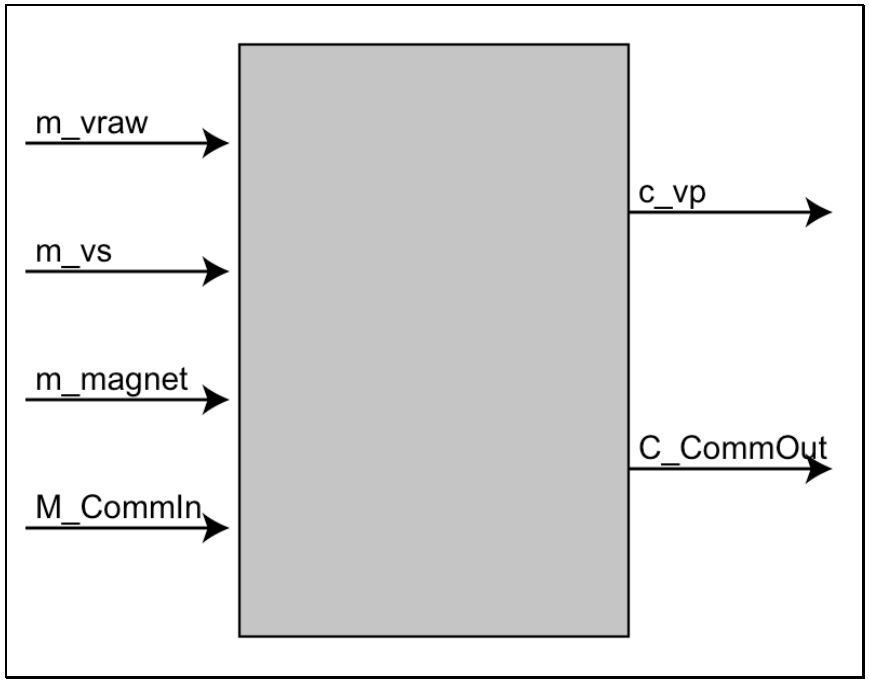


Figure 1. Context Diagram

2.1. Monitored Variables

Name	Units/Type	Device	Range	Description
M_CommIn	None	Serial DB-9	-	Serial communications input packet
m_magnet	y_magnet	External	-	Determines whether magnet is in place
m_vraw	mV	Ventricle Co-ax	0.0 – 5000.0	Ventricular output signal after filtering
m_vs	boolean	Ventricle Co-ax	{True, False}	Ventricle sense after filtering and above threshold

2.2. Controlled Variables

Name	Units/Type	Device	Range	Description
C_CommOut	None	Serial DB-9	-	Serial communications output packet
c_vp	mV	Ventricle screw terminals	0.0 – 12000.0	Ventricle pace output between ring and tip

3. Constants

3.1. Programmable Parameters

Name	Units/Type	Description	Value / Range / Tol
p_pacingState	y_pacingState	Pacing state	PERMANENT
p_pacingMode	y_pacingMode	Pacing mode	VVI
p_hysteresis	boolean	True if hysteresis is to be used, False if no hysteresis	False / True - False
p_hysteresisInterval	mSec	Additional delay interval used when hysteresis is included	300 / 200 – 500 / ±4
p_lowrateInterval	mSec	Delay interval that specifies maximum delay after a ventricle pace without a spontaneous sense or another pace	1000 / 343 – 2000 / ±8
p_vPaceAmp	mV	Desired amplitude of a ventricular pace	3500 / 500 – 7000 / ±12%
p_vPaceWidth	mSec	Desired width of a ventricular pace	0.4 / 0.1 – 1.9 / 0.2
p_VRP	mSec	Duration of ventricular refractory period	320 / 150 – 500 / ±8

3.2. Internal Constants

Name	Units/Type	Description	Value / Range / Tol
k_egram	byte	Function code indicating request to transmit egram data	‘47’h
k_echo	byte	Function code indicating request to transmit programmable parameters to DCM	‘49’h
k_estop	byte	Function code indicating that transmission of egram data should stop	‘62’h
k_pparams	byte	Function code indicating transmission of programmable parameter values	‘55’h
k_soh	byte	Expected second byte in transmissions (Start Of Header)	‘01’h
k_streamPeriod	mSec	Period at which to stream data from Pacemaker to DCM	4.0 / 1.0 – 5.0 / 0.1
k_sync	byte	Expected first byte in transmissions (Synchronization)	‘16’h

The shaded cell indicates a relaxation of the original requirement for this project.

4. VVI Mode in Permanent Pacing

Assumptions:

- p_pacingMode = VVI **AND** p_pacingState = PERMANENT
- signals are detected on their rising edge
- p_vPaceWidth < p_VRP < p_lowrateInterval

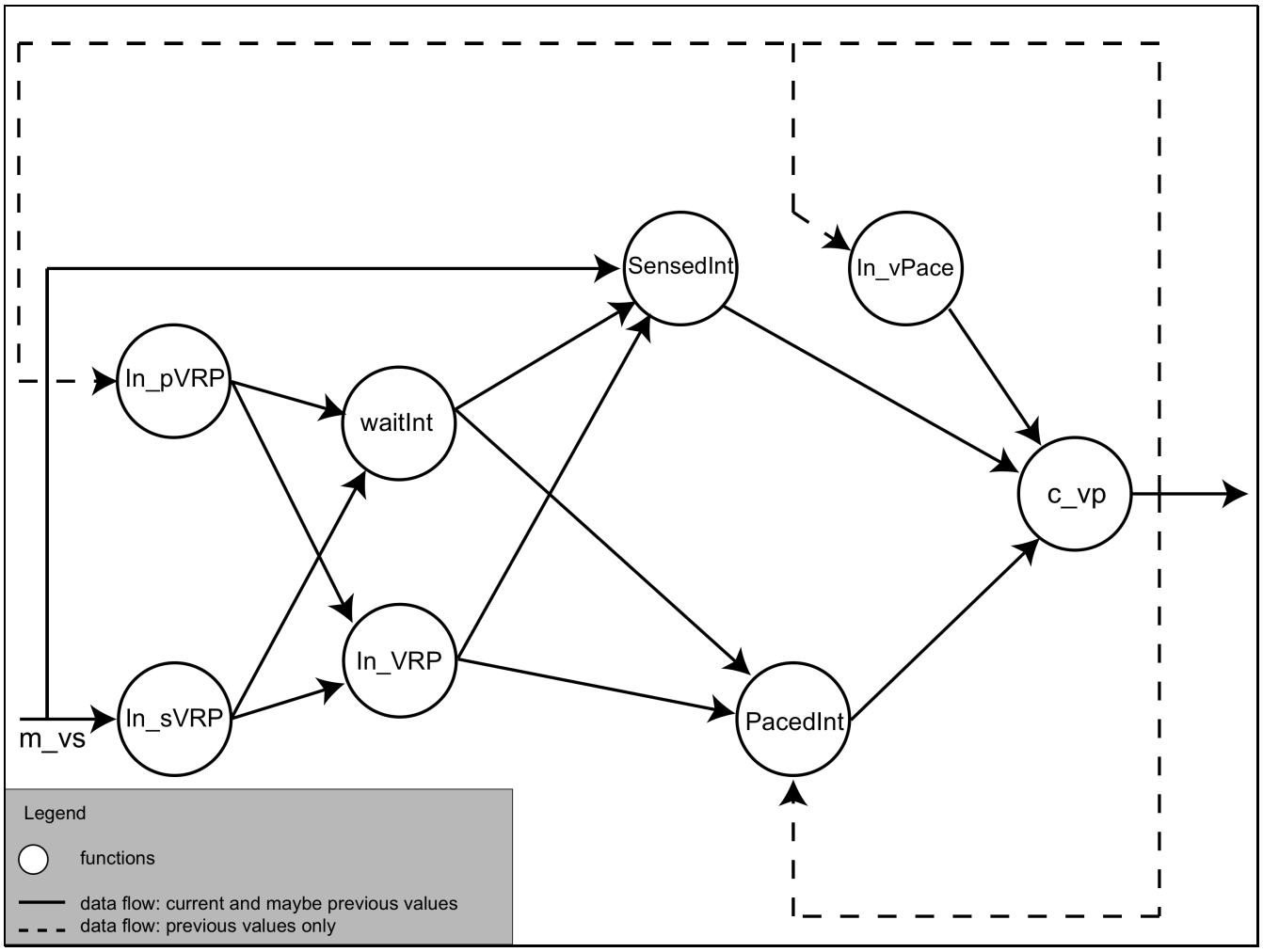


Figure 2. Data Flow for VVI Mode

waitINT = f_waitInterval
SensedInt = Spontaneous ventricle sense in wait interval
PacedInt = Ventricle paced in wait interval

4.1. Ventricular Refractory Periods

The refractory period is the time immediately after a sense or pace, in which a new sense or pace must be ignored. In our system there is no difference between sense and pace ventricular refractory periods (VRPs). However, we have specified different functions for each in case this changes in the future.

4.1.1 Sense VRP

Determines whether the VRP is active or not after a spontaneous ventricular sense.

4.1.1.1 Output Units/Type and Initial Value

Name	Units/Type	Initial Value
In_sVRP	boolean	false

4.1.1.2 Definition

$In\_sVRP = t_{now} - tm(NOT\ In\_sVRP_{-1}\ \&\ m\_vs\ \&\ NOT\ m\_vs_{-1}) \leq p\_VRP$

4.1.2 Pace VRP

Determines whether the VRP is active or not after a ventricular pace.

4.1.2.1 Output Units/Type

Name	Units/Type	Initial Value
In_pVRP	boolean	false

4.1.2.2 Definition

$In\_pVRP = t_{now} - tm(NOT\ In\_pVRP_{-1}\ \&\ c\_vp_{-1} = p\_vPaceAmp\ \&\ c\_vp_{-2} = 0) \leq p\_VRP$

4.1.3 VRP

Determines whether any VRP is active or not after a ventricular pace.

4.1.3.1 Output Units/Type

Name	Units/Type
In_VRP	boolean

4.1.3.2 Definition

In_VRP = In_sVRP OR In_pVRP

4.2. Ventricular Pace Width

Controls the width of the ventricular pace.

4.2.1 Inputs

Name	NL Expression	Reference
c_vp_{-1}	-	4.6.4

4.2.2 Output Units/Type

Name	Units/Type
In_vPace	boolean

4.2.3 Definition

$$In\_vPace = t_{now} - tm(c\_vp_{-1} = p\_vPaceAmp \;\&\; c\_vp_{-2} = 0) \leq p\_vPaceWidth$$

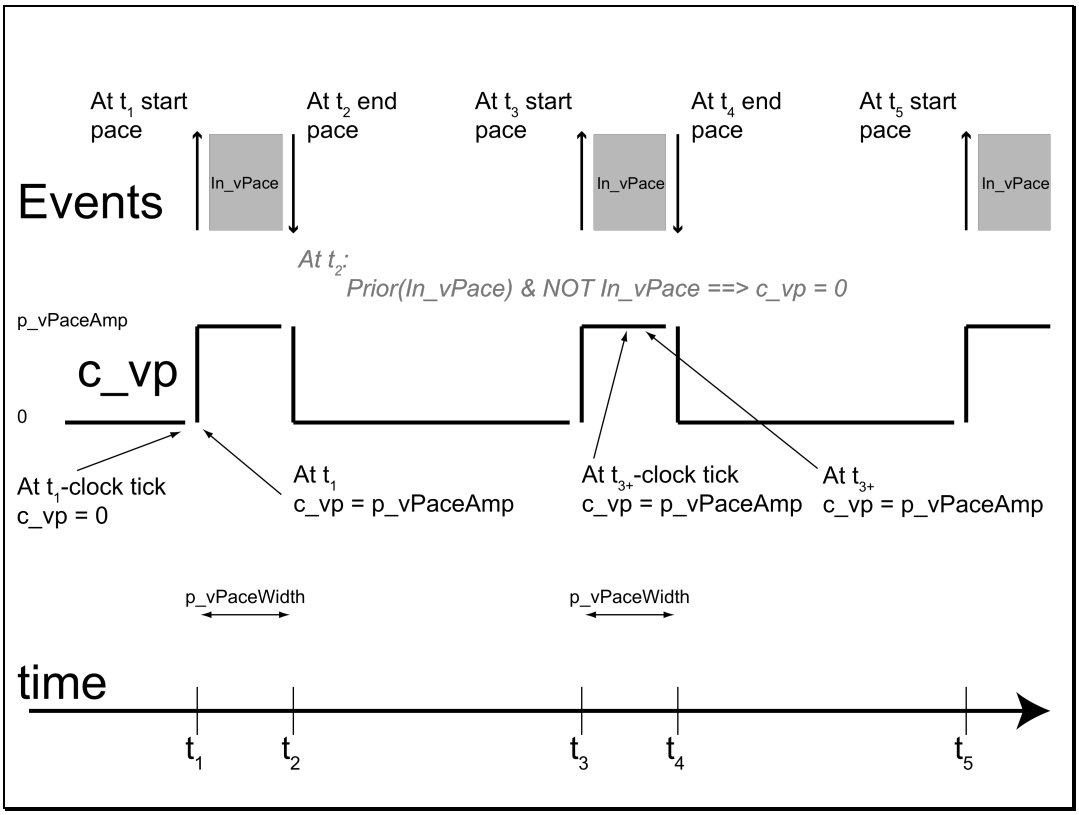


Figure 3. Explanation of In_vPace

4.3. Wait Interval

Determines the time interval to wait after a ventricular event. The wait intervals may be different for spontaneous sense events compared with paced events.

4.3.1 Inputs

Name	NL Expression	Reference
-	In_sVRP	4.1.1.2
-	In_pVRP	4.1.2.2

4.3.2 Initial Values

None.

4.3.3 Output Units/Type

Name	Units/Type
f_waitInterval	mSec

4.3.4 f_waitInterval

		Result
	Condition	**f_waitInterval**
NOT Prior(In_sVRP) & In_sVRP	p_hysteresis	p_lowrateInterval + p_hysteresisInterval
	NOT p_hysteresis	p_lowrateInterval
NOT Prior(In_pVRP) & In_pVRP		p_lowrateInterval
[Prior(In_sVRP) OR NOT In_sVRP] & [Prior(In_pVRP) OR NOT In_pVRP]		No Change

4.4. Spontaneous ventricle sense in wait interval

Determines whether or not there is currently a spontaneous sense within the appropriate maximum time interval.

4.4.1 Inputs

Name	NL Expression	Reference
f_waitInterval	-	4.3.4
-	In_VRP	4.1.3.2
m_vs	-	2.1

4.4.2 Initial Values

None.

4.4.3 Output Units/Type

Name	Units/Type
Spontaneous ventricle sense in wait interval	boolean

4.4.4 Definition

	Spontaneous ventricle sense in wait interval
(NOT In_VRP & NOT m_vs) Held For (f_waitInterval – p_VRP)	False
NOT [(NOT In_VRP & NOT m_vs) Held For (f_waitInterval – p_VRP)]	True

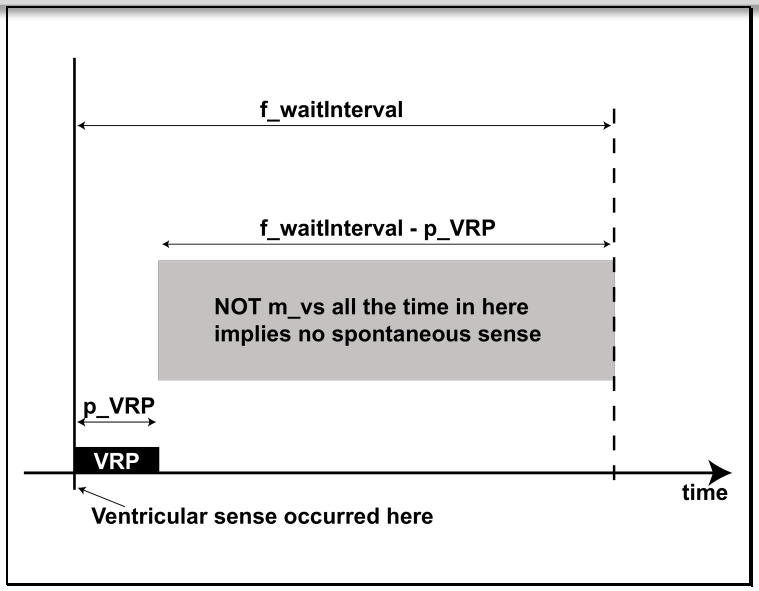


Figure 4. Spontaneous Ventricular Event

4.5. Ventricle paced in wait interval

Determines whether or not there is currently a ventricular pace within the appropriate maximum time interval.

4.5.1 Inputs

Name	NL Expression	Reference
c_vp_{-1}	-	4.6.4
f_waitInterval	-	4.3.4
-	In_VRP	4.1.3.2

4.5.2 Initial Values

None.

4.5.3 Output Units/Type

Name	Units/Type
Ventricle paced in wait interval	boolean

4.5.4 Definition

	Ventricle paced in wait interval
(NOT In_VRP & $c\_vp_{-1} = 0$) Held for (f_waitInterval – p_VRP)	False
NOT [(NOT In_VRP & $c\_vp_{-1} = 0$) Held for (f_waitInterval – p_VRP)]	True

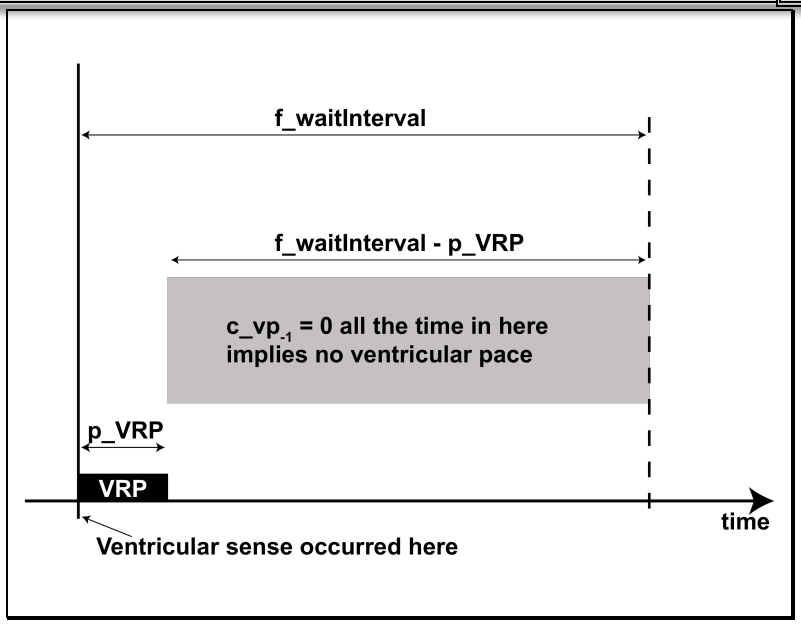


Figure 5. Paced Ventricular Event

4.6. VVI Mode Ventricle Pace

Determines the value of the Ventricular Pacing Output between ring and tip of the board in VVI mode.

4.6.1 Inputs

Name	NL Expression	Reference
-	In_vPace	4.2.3
-	Spontaneous ventricle sense in wait interval	4.4.4
-	Ventricle paced in wait interval	4.5.4

4.6.2 Initial Value

Name	Value	Reference
c_vp_{-i}, i=1,2	0	-

4.6.3 Output Units/Type

Name	Units/Type
c_vp	mV

4.6.4 c_vp

Condition	*Result* **c_vp**
NOT [Spontaneous ventricle sense in wait interval] & NOT [Ventricle paced in wait interval]	p_vPaceAmp
Prior (In_vPace) & NOT In_vPace	0
[Spontaneous ventricle sensed in wait interval OR Ventricle paced in wait interval] OR NOT [Prior(In_vPace)] OR In_vPace	No Change

5. Pacemaker Interface with Device-Controller Monitor (DCM)

5.1. Communications from DCM to Pacemaker

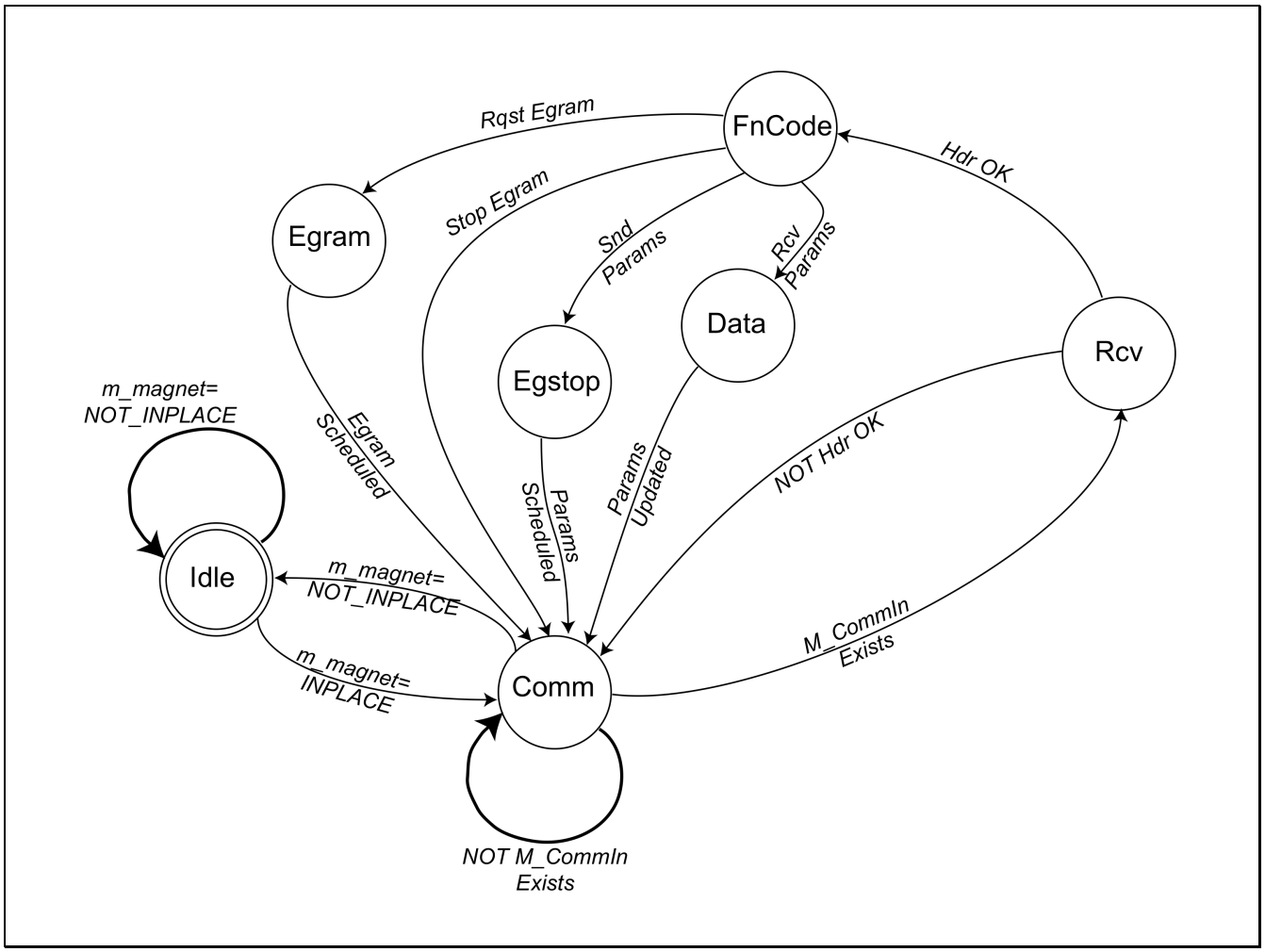


Figure 6. Pacemaker Receive

Egram Scheduled	=	Transmission of egram to DCM triggered
Params Scheduled	=	Transmission of programmable parameters to DCM triggered
Params Updated	=	Values of programmable parameters updated from communications packet, if Data checksum was OK and parameters pass validity checks - else Data checksum was not OK or validity failed and no action
Rcv Params	=	i_FnCode = k_pparams
Snd Params	=	i_FnCode = k_echo
Rqst Egram	=	i_FnCode = k_egram
Stop Egram	=	i_FnCode = k_estop
Hdr OK	=	i_CommIn_0 = k_sync & i_CommIn_1 in [k_echo, k_pprams, k_egram, k_estop]

Initial state is Idle.

5.1.1 Structure of Receive Packet (i_CommIn)

Item	Size (bytes)	Start Byte Number
SYNC	1	0
SOH	1	
FnCode	1	1
Header ChkSum	1	
$Data_{i,}$ i=1,..,n	n	2
Data ChkSum	1	2 + n

where n = 13. Total number of bytes in receive packet is n + 3.

5.1.2 Data contents of i_CommIn

If FnCode = k_pparams:

Name	Size (bytes)	Start Byte Number
p_pacingState	1	0
p_pacingMode	1	1
p_hysteresis	1	2
p_hysteresisInterval	2	3
p_lowrateInterval	2	5
p_vPaceAmp	2	7
10*p_vPaceWidth	2	9
p_VRP	2	11
spare	5	

If FnCode ≠ k_pparams:
$Data_i$ = 0, i=1,..,n; Data ChkSum = 0.

The shaded rows show elements that we would normally include. To reduce the number of bytes in the package, we have removed them from this version. The same is true in sections 5.1 and 5.2.

5.2. Communications from Pacemaker to DCM

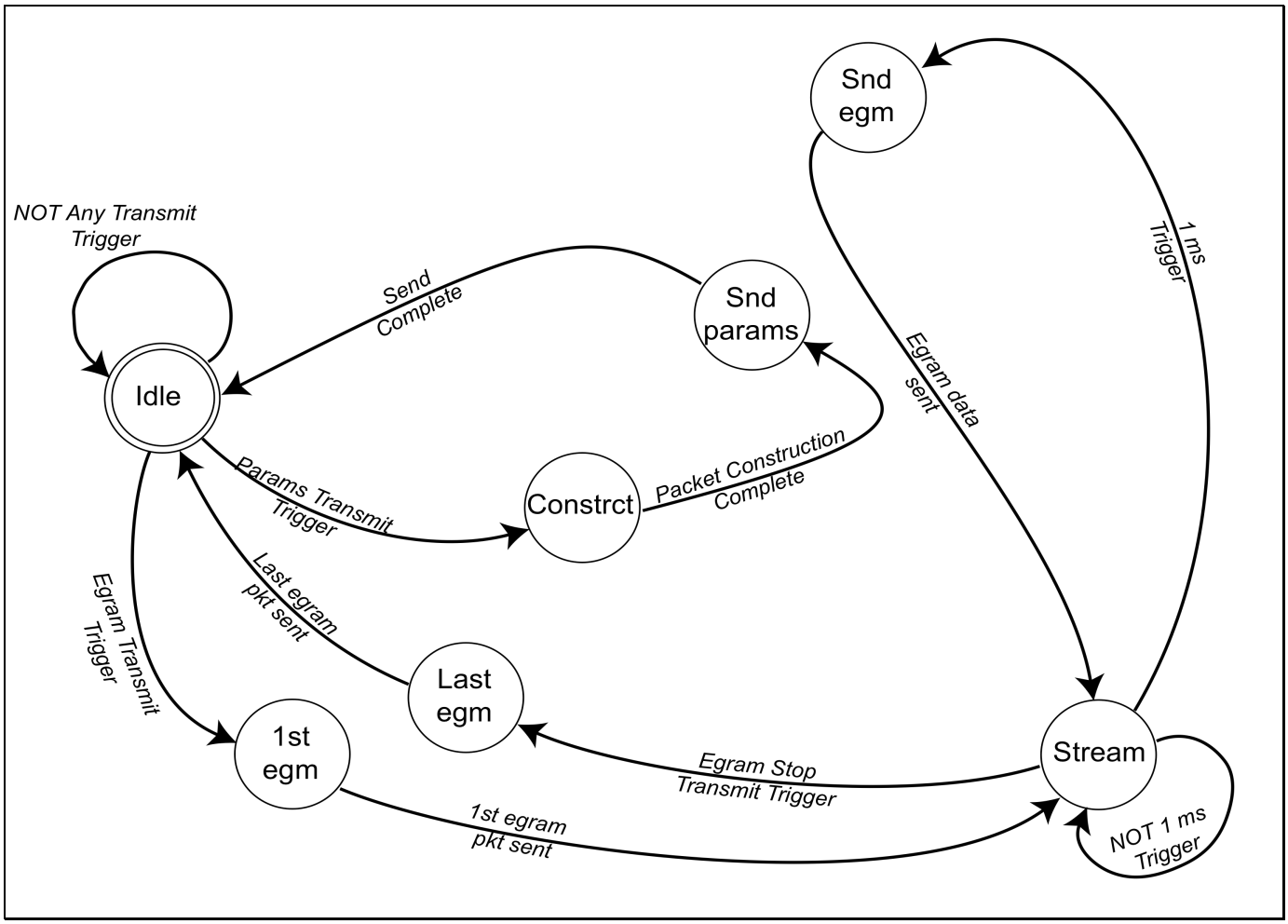


Figure 7. Pacemaker Transmit

1st egram pkt sent	=	Construct and transmit 1st packet of egram data
1 ms Trigger	=	SyncPeriodic(k_streamPeriod) (still called 1 ms but relaxed in this SRS)
Any Transmit Trigger	=	i_CommIn_1 in [k_echo, k_egram, k_stop] and output triggered
Egram data sent	=	Finished sending ventricle signal and marker for current milli-second
Egram Stop Trigger	=	$i\_CommIn_1 = k\_estop$
Egram Transmit Trigger	=	$i\_CommIn_1 = k\_egram$
Last egram pkt sent	=	Construct and transmit last packet of egram data
Packet Construction Complete	=	Finished construction of parameter transmit packet
Params Transmit Trigger	=	$i\_CommIn_1 = k\_echo$
Send Complete	=	Finished sending parameters output packet

Initial state is Idle.

5.2.1 Structure of Transmit Packet (o_CommOut)

Item	Size (bytes)	Start Byte Number
SYNC	1	0
SOH	1	
FnCode	1	1
Header ChkSum	1	
$Data_{i,}$ i=1,..,n	n	2
Data ChkSum	1	2 + n

Data ChkSum = f_chk($Data_1$, $Data_2$, …, $Data_n$)
n = 13. Total number of bytes in receive packet is n + 3.

5.2.2 Data contents of o_CommOut when not streaming egram data

i_CommIn_1 = k_echo: FnCode = k_pparams.

Name	Size (bytes)	Start Byte Number
p_pacingState	1	0
p_pacingMode	1	1
p_hysteresis	1	2
p_hysteresisInterval	2	3
p_lowrateInterval	2	5
p_vPaceAmp	2	7
10*p_vPaceWidth	2	9
p_VRP	2	11
spare	5	

5.2.3 Data contents of o_CommOut, i_CommIn_1 in [k_egram, k_estop]

i_CommIn_2 = k_egram	i_CommIn_2 = k_estop
FnCode = k_egram $Data_1$: $Data_2$ = m_vraw $Data_3$: $Data_4$ = f_marker $Data_i$ = 0, i=5,..,13 Data ChkSum = f_chk($Data_1$, $Data_2$, …, $Data_n$), n = 13.	FnCode = k_estop $Data_i$ = 0, i=1,..,13 Data ChkSum = f_chk($Data_1$, $Data_2$, …, $Data_n$), n = 13.

5.2.4 o_CommOut Transmission while streaming data

This occurs when egram data has been started and has not yet been stopped.

Item	Size (bytes)	Start Byte Number
m_vraw	2	0
f_marker	2	2

5.3. Marker value to be transmitted

The marker is constructed just in time for the information to be sent, governed by the 1ms timer specified in Figure 7.

5.3.1 Inputs

Name	NL Expression	Reference
-	In_sVRP	4.1.1.2
-	In_pVRP	4.1.2.2

5.3.2 Output Units/Type

Name	Units/Type
f_marker	char : char

5.3.3 f_marker

		f_marker
In_sVRP	NOT Prior(In_sVRP)	'VS'
	Prior(In_sVRP)	'()'
In_pVRP	NOT Prior(In_pVRP)	'VP'
	Prior(In_pVRP)	'()'
NOT In_sVRP & NOT In_pVRP		'--'

5.4. Effect on Pacing

When the magnet is recognized as being in place, pacing behaviour is also affected. This is specified in the following functions.

5.4.1 Magnet Effect on Pacing

When the magnet is in place, pacing is VOO, otherwise VVI.

5.4.1.1 Inputs

Name	NL Expression	Reference
m_magnet	-	2.1
-	In_vPace	4.1.2.2

5.4.1.2 Output Units/Type

Name	Units/Type
c_vp	mV
f_vooInterval	p_lowrateInterval
f_vPaceWidth	p_vPaceWidth

5.4.1.3 c_vp, Duration Interval and Pulse Width

	c_vp	f_vooInterval	f_vPaceWidth
m_magnet = INPLACE	See 5.4.2.3	No Change	No Change
m_magnet = NOT_INPLACE	See 4.6.4	p_lowrateInterval	p_vPaceWidth

5.4.2 Ventricular Pace in VOO Mode

5.4.2.1 Inputs

Name	NL Expression	Reference
m_magnet	-	2.1

5.4.2.2 Output Units/Type

Name	Units/Type
c_vp	mV

5.4.2.3 c_vp

	c_vp
Periodic(magEvent, f_vooInterval)	p_vPaceAmp
Prior(In_vPace) & NOT In_vPace	0
NOT Periodic(magEvent, f_vooInterval) OR NOT Prior(In_vPace) OR In_vPace	No Change

where magEvent ≡ (m_magnet = INPLACE) & (NOT In_VRP)

6. Requirements for the Device-Controller Monitor

TBD. The DCM requirements are obviously described in the Boston Scientific requirements document. For this project you need to:

1. Enable the user to download programmable parameters to the Pacemaker.
2. Request to see current parameters in the Pacemaker – this will trigger a communications packet from the Pacemaker.
3. Request that the Pacemaker stream egram data which must then be displayed and printed if the user requests that.
4. Instruct the Pacemaker to stop streaming the egram data.
5. Store and display/print a history of programmable parameters in the Pacemaker.
6. Details of the communication protocol can be deduced from the Pacemaker communications.

7. Performance Requirements

1. There is an over-riding requirement to put the processor in standby mode whenever possible. This is to preserve battery power in the "real" pacemaker.

2. Serial communication baud rate shall be 57600.

8. Anticipated Changes and Changes Not Likely

Likely changes:

1. Need to set programmable resistor values to achieve desired ranges and accuracy.
2. Accommodate additional modes, including atrium sensing and pacing.
3. Comply with the histogram requirements in [2].
4. Comply with rate smoothing requirements in [2].

Unlikely changes:

1. The number of leads to attach to the heart is not likely to change.

9. Input Variables

Name	Address	Format	Transfer Event	M - I Mapping	Tol.	Ref.
i_CommIn	RCREG	byte			-	
i_magnet	PORTB, RB7	bit			-	
i_vraw						
i_vs				INT0=1 ➔sensed		

10. Output Variables

Name	Address	Format	Transfer Event	C - O Mapping	Tol.	Ref.
o_CommOut	TXREG	1 byte			-	
o_vp						

11. M-I Mappings and Transfer Events

Code provided.

12. C-O Mappings and Transfer Events

Code provided.

13. Generic Notation and Functions

13.1 The requirements model is a finite state machine with discrete time and an arbitrarily small clock-tick. For variables, name represents its value at the current time, and $name_{-1}$ represents its value at the previous clock-tick, etc.

13.2 t_{now} is the current time.

13.3 tm(Condition) returns the most recent time that Condition changed from False to True.
Initial value: tm(Condition) = 0 at $t_{now} = 0$

	tm(Condition)
NOT $Condition_{-1}$ & Condition	t_{now}
$Condition_{-1}$ OR NOT Condition	No Change

13.4 (Condition) Held For (d), with tolerances -δL, +δR on the duration d.

	duration	**Event_start_time**
(Condition = True) & ($Condition_{-1}$ = False)	Any value in [d-δL, d+δR]	t_{now}
(Condition = False) OR ($Condition_{-1}$ = True)	No Change	No Change

At time of initialization, duration is any value in [d-δL, d+δR], and $Event\_start\_time_{-1} = 0$.

		(Condition) Held for (d)
Condition = True	t_{now} – Event_start_time ≥ duration	True
	t_{now} – Event_start_time < duration	False
Condition = False		False

13.5 $f\_Chk(a_0, a_1, a_2, .., a_n) =$
a_0 XOR a_1 XOR a_2 XOR ... XOR a_n

13.6 Periodic(cond, d), with tolerances -δL, +δR on the duration d.
This function (Periodic) is True for 1 clock tick at the instant that the Condition changes from False to True, and, as long as Condition remains True, the function is True again, a duration "period" after the most recent time it changed from False to True. Initially, $Periodic_{-1}$ is assumed to be False.

	period
$Periodic_{-1}$ = True	Any value in [d-δL, d+δR]
$Periodic_{-1}$ = False	No Change

			Periodic	previous_ pulse_time
Condition = True	$Condition_{-1}$ = False		True	t_{now}
	$Condition_{-1}$ = True	$t_{now} \geq previous\_pulse\_time_{-1} + period$	True	t_{now}
		$t_{now} < previous\_pulse\_time_{-1} + period$	False	No Change
Condition = False			False	No Change

13.7 SyncPeriodic(d) with tolerances -δL, +δR on the duration d.
This function (SyncPeriodic) is True for 1 clock tick every time period d, but some tolerance is allowed on d. However, the tolerance does not accumulate.

Initially, n = 0, Δ = any value in [0, δR], $SyncPeriodic_{-1}$ = False.

	Δ	n
$SyncPeriodic_{-1}$ = True	Ant value in [-δL, δR]	n + 1
$SyncPeriodic_{-1}$ = False	No Change	No Change

	SyncPeriodic
$t_{now} \geq n \cdot d + \Delta$	True
$t_{now} < n \cdot d + \Delta$	False

13.8 Hex values are represented as ‘dd’h where d is a hex digit in the range 0-9 A-F.

14. References

[1] Cliff Nixon, James Smith, Tony Ulrich, Rebecca Davis, Christopher Larson, Kua Cha, *Academic Dual Chamber Pacemaker*, University of Minnesota, 2006.

[2] *PACEMAKER System Specification*. Boston Scientific. 2007.

[3] Barold, Stroobandt and Sinnaeve, *Cardiac Pacemakers Step by Step, An Illustrated Guide,* Blackwell Publishing Inc. 2004