This is a guide only. Difficult to predict how groups will partition the design between draft system design and component design.

Criteria	Mark	Out of
Title page, incl. names and Ids	✓	1
Table of Contents	✓	1
List of Figures	✓	1
List of Tables	✓	1
All pages are numbered	✓	1
Every figure has a caption and every table has a heading	✓	1
Revision History	✓	1
Supporting Material – appendices, etc.	✓	1
Introduction – Scope, purpose, background, roadmap		7
Spelling and grammar	✓	10
Style – paragraph structure, concise ideas, flow between sections, navigation references, subsections logically organized		10
Hyperlinking of references		5
Module Guide		5
Decomposed to small enough modules; modules are not too small (larger than a single function); modules are not too big; when a module is decomposed, it is decomposed into more than one module		5

Decomposition by secrets, not by sequence of steps, are the secrets realistic? are they really secrets? are they nouns, not verbs?		5
Feasible design		5
Flexible Design		5
Connection between requirements and design – division into hovercraft and base station, hardware versus software, abstract to concrete.		2
Numbered list of anticipated changes at design level		2
Numbered list of unlikely changes – language of interface, hardware platform, division between hovercraft and base station, etc.		2
System decomposition tree		2
Hardware hiding, behaviour hiding, software decision hiding modules		2
Secrets sum to previous level of the tree?		2
Description of each module using a consistent template - well-formed, one module – one secret, same secret does not appear in multiple modules		2
Traceability for requirements to modules		5
Traceability for anticipated changes to modules - is this matrix sparse? Is it		5

reasonable? Does each anticipated change have at least one module associated with it?		
Uses hierarchy – not a control flow diagram		5
Hardware/Software decomposition identified		5
Hardware schematics and drawings included as needed		20
MIS – MID		
exported constants section is present (if appropriate)		5
exported types section is present (if appropriate)		5
exported functions section is present (if appropriate)		5
state variables are identified with their types		5
state invariants are identified		5
assumptions are listed		5
access routine semantics are included for all access programs		5
set access routines have a transition and an exception entry		5
get access routines have an output and an exception entry		5

set-get access routine entries have a transition-output and an exceptions entry		5
module purpose is clear		5
Check one mathematical expression for correctness		5
Check a second mathematical expression for correctness		5
Consistent - is there is deviation from the team's conventions?		3
Are obvious exceptions listed?		3
Check access routine to see if the exception behaviour is deterministic		5
Check access routines to see if when the exceptions entry does not indicate an exception, the transition, output, or transition-output entry provides a normal-case behaviour		5
Overall impression 1: Understandable? All error cases covered? Interface is usable? Hides its secret?		5
Overall impression 2: Understandable? All error cases covered? Interface is usable? Hides its secret?		5
Module does something – i.e., Update and/or Output		5
Overall impression: Document is complete, clear and concrete enough that a programmer		40

or a hardware technician can code/create the software modules/system component.		
Totals:		255

Feedback: