Rev. Level	ECO	Approved By	Date	Revision Description
Rev 9		ECO	09/17/2019	

System Level Indiana-C
Engineering Requirements Specification

Document 099-16082
Revision 9
September 17, 2019

Authors:

Likai Li likai_li@apple.com

Table of Contents

Change History

Version	Description	Date	By
1	- Copied from IN-A - Added GCOL testing	04/05/19	Likai Li
2	- Update GCOL PFL trends	04/05/19	Likai Li
3	- Updated NVM plant code	04/25/19	Likai Li
4	- Updated black level offset command - Updated Grayspot image region	05/02/19	Likai Li
5	- Cleaned up unused ERS trends - Added notes for SFR delta - Added PFL trends - Added GCOL trends - Update EEEER table and NVM check	06/18/19	Likai Li
6	- Added PFL testing coverage	06/21/19	Likai Li
7	- Updated GCOL AF pos spec - Updated EEEER table for new configs	06/28/19	Likai Li
8	- Updated rotation spec in IQC - Updated PFL spec in CCB	07/11/19	Likai Li
9	- Update AF Pos spec - Updated grayspot crop region - Rolled back PFL spec in CCB - Updated EEEER table and NVM check for EVT	09/17/19	Likai Li

About This Document

This document describes the Rear Camera System used in the Apple Indiana-A project. The contents of this Engineering Requirements Specification, including any Apple-Vendor project-specific information, are **Apple Confidential Information** subject to the non- disclosure and use restrictions set forth in the Confidentiality Agreement between Vendor and Apple.

No part of this specification and its contents may be shared, distributed or disclosed, in any form, to any third party without explicit written agreement from the Apple DRI. This restriction includes the sharing of information by Vendor to its suppliers or customers without the prior written consent by Apple. Where possible, Apple will always try to directly distribute the specifications to the Vendor's customers or suppliers who need them as this is the preferred means of document distribution.

Additionally, requests from a Vendor's customers for information or for changes related to any specifications or process items should be referred to the appropriate Apple DRI for a response. Any audit access requests by the Vendor's customers in the Apple supply chain related to project-specific specifications or processes should be refused to ensure protection of Apple Confidential Information. However, the Vendor's customers can request, through the Apple DRI, that these specifications or process be audited by Apple personnel.

If any request is made by a third party contrary to the above requirements, such request should be immediately escalated to the appropriate Apple DRI for review and resolution. For the purposes of this specification, Vendor is defined as the company supplying the part/ service specified by this document.

Audience

This guide is for Apple internal use only. The people who benefit from this guide are:

- Engineers who are designing components of the camera system.
- Engineers who are designing electrical or mechanical parts interfacing to or related to the camera system.
- Engineers who are responsible for the testing and validation of the camera system.

Related Documents

Table 1: Related Documents

Specification	Description
Indiana Camera Documents	613-11183: IN Module MCO
	056-08304: MCO, FLEX, IN-C
	12.0cm Matrix SFR Target Ver.K.1.3 - 73.03° DFOV
	55.0cm Matrix SFR Target Ver.K.1.2 - 73.926° DFOV
	12.0cm Circle SFR Target Ver.IN-A C5.0 - 73.03° DFOV
	55.0cm Circle SFR Target Ver.IN-A C5.0 - 73.926° DFOV
	099-12589: ERS,Cosmetic Criteria,RCAM,IN

Part Identification

Part Numbers

APN	Integ	Plant Code	Substrate-Actuator-Lens-Flex	NVM Substrate [0x07]	NVM Actuator [0x09]	NVM Lens [0x0A]	NVM Flex [0x0E]	EEEER	Config
651-00202	LGIT	DN8	Kyocera-ALPS-Largan-Fujikura	1	2	1	8	LV401	C3001 C3003 C3091 C3092 C3093 C3094
			Kyocera-ALPS-Largan-Mektec	1	2	1	2	LV402	C3002 C3040 C3095 C3096 C3097 C3098

If for any reason, vendor ships parts that come out of a new location, or a configuration different from that listed above, a new Plant Code and EEEER config code will need to be provided by Apple. Changes may not be made without these new codes.

The Plant Code and EEEER code shall be used to determine a unique serial number for each part, which shall be in the form:

PPPYWWDSSSSEEEERV

Where PPP is the plant code, YWWD is the date of manufacture at supplier, SSSS is the sequence number, and EEEER is the config code, and V is the SN checksum.

Mechanical and Cosmetic IQC

IQC Cosmetic Inspection Criteria

No cosmetic defects that could affect functionality or exceed MCO dimensions are allowed.

Examples:

a. Contamination on lens

b. Digs or nicks in FPC that could have caused a crack in a trace

c. B2B pin or connector deformation which could prevent good electrical connection.

d. Gap between FPC and Stiffener

Detailed cosmetic specs are in **cosmetic inspection criteria document.**

System Factory Functional Test Specifications

The following are system level factory test specifications for the Indiana-A Rear Camera. All test items below must be performed on 100% of systems.

There are two types of tests required: NVM integrity check, imaging tests. The NVM integrity check should be performed at IQC and FATP to confirm values programmed in the NVM are within acceptable limits. The imaging tests should be performed at IQC and FATP in order to exercise the imaging performance of the camera module in system.

For imaging tests, the hxisp command line tool should be used to capture images. Images should be captured in 420 format, with ISP sharpening disabled. Temporal Noise Reduction through frame averaging may be applied for SFR image

IQC/FATP/DQE Test Parameters

Test Type	Input	Value	Notes
Blemish	ROI Size	40 x 30	W x H, Pixels
	Inner [W H]	[50% 50%]	Inner %
	Normalization	N/A	
	Normalization Target	N/A	
Grayspot	Subsample Rate	16, 16	Row, Col
	Filter Width	20 x 20	Pixels
	Col/Row MaxRateDiff	Max 0.7	
	Col/Row MaxRate	Max 1.0	
Color Uniformity	ROI Size	108 x 108	W x H, Pixels
	Border Size	TBD	Pixels
Relative Uniformity	Block Size	12 x 9	W x H, Pixels
	Border Size	5	Blocks
	Crop ROIs	0	
Relative Illumination	Block Size	40 x 30	W x H, Pixels
MTF	ROI Size	101 x 101	W x H, Pixels
	Frame Averaging	10 frames	TNR
LCB	ROI Size	N/A	W x H, Pixels
	Filter Width	N/A	Pixels
CTF	ROI Size	101 x 101	W x H, Pixels
Row Noise	Periodicity Weighting	0.5	-

IQC/FATP/DQE Imaging Test Coverage

Test Type	Test Item	Units	IQC		FATP		DQE		Notes
			Min	Max	Min	Max	Min	Max	
Blemish	Max Ratio	%	0	23	0	24	0	24	
	Count	Count	0	2	0	2	0	2	
	Region_Center	Count	0	0	0	0	0	0	
	Region_Outer	Count	0	2	0	2	0	2	
	MaxSizeCenter	Count	0	0	0	0	0	0	
	MaxSizeOuter	Pixels	0	1	0	1	0	1	
	Luminance	DN	50	200	50	200	50	200	
Grayspot	Gray Spot Cluster Count	Count	0	0	0	0	0	0	IQC crop to 4096x3072 FATP crop to 4032x3024
Color Uniformity	CU	%	0	8.5	0	9	0	9	
Relative Uniformity	Group Count	Count	0	0	0	0	0	0	
	RU Center	%	0	10	0	10	0	10	
	RU Edge	%	0	10	0	10	0	10	
	RU Corner	%	0	10	0	10	0	10	
Relative Illumination	Relative Illumination	-	0.8	1	0.78	1	0.78	1	
	RI Center X	-	-1	1	-1	1	-1	1	
	RI Center Y	-	-1	1	-1	1	-1	1	
Defective Line "linetest"	Max Rate Row Y	DN	0	0.2	0	0.2	0	0.2	
	Max Rate Col Y	DN	0	0.2	0	0.2	0	0.2	
	Max Rate Row Cr	DN	0	0.2	0	0.2	0	0.2	
	Max Rate Col Cr	DN	0	0.2	0	0.2	0	0.2	
	Max Rate Row Cb	DN	0	0.2	0	0.2	0	0.2	
	Max Rate Col Cb	DN	0	0.2	0	0.2	0	0.2	
Defective Line (DARK)	Max Rate Row Y	DN	0	6	0	6	0	6	
	Max Rate Col Y	DN	0	4	0	4	0	4	
	Max Rate Row Cr	DN	0	0.6	0	0.6	0	0.6	
	Max Rate Col Cr	DN	0	0.2	0	0.2	0	0.2	
	Max Rate Row Cb	DN	0	0.6	0	0.6	0	0.6	
	Max Rate Col Cb	DN	0	0.2	0	0.2	0	0.2	
Row Noise Ratio	TempRrR_Total	ratio	14	-	16	-	16	-	Apply settings: LCD On, PSRR On. RnR calculated on Gr (G1) channel only.
	RnR_Summation	ratio	13	-	15	-	15	-	
	RnR_Periodicity	ratio	-	-	-	-	-	-	
SFR 12cm (70lp/mm)	Luminance	DN	50	200	50	200	50	200	
	X Tilt	Deg.	-1.4	1.4	-2.3	2.3	-2.3	2.3	
	Y Tilt	Deg.	-1.4	1.4	-2.5	2.5	-2.5	2.5	
	Total Tilt	Deg.	-	2.5	-	3.3	-	3.3	
	Rotation	Deg.	-1.0	1.0	-2.5	2.5	-2.5	2.5	
	DFOV	Deg.	71	75	71	75	71	75	
	12cm AFPos GCOL	-	158	238	156	240	148	250	GCOL
	12cm zPos	um	Data collection						
	12cm zDeltaPos	um	Data collection						
	Circle Center SFR	N/A	0.70	1	0.68	1	0.58	1	15% Degradation
	Circle 0.3F SFR sag/tan	N/A	0.60	1	0.58	1	0.49	1	
	Circle 0.6F SFR sag/tan	N/A	0.36	1	0.34	1	0.29	1	
	Circle 0.85F SFR tan	N/A	0.32	1	0.3	1	-	1	
	Circle SFR 0.3F Delta	N/A	-	-	-	-	-	-	
	Circle SFR 0.6F Delta	N/A	-	-	-	-	-	-	

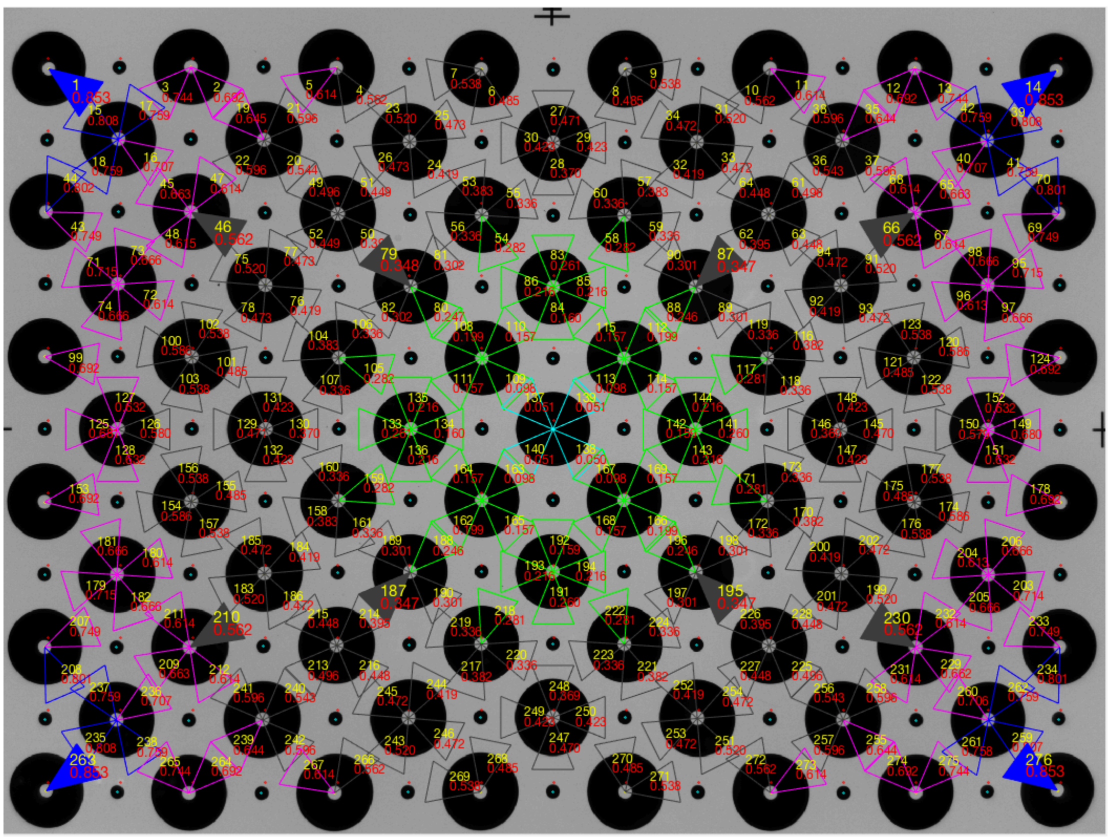
Test Type	Test Item	Units	IQC		FATP		DQE		Notes
			Min	Max	Min	Max	Min	Max	
	Circle SFR 0.85F Delta	N/A	-	-	-	-	-	-	
SFR 55cm (70lp/mm)	Luminance	DN	50	200	50	200	50	200	
	X Tilt	Deg.	-1.3	1.3	-1.7	1.7	-1.7	1.7	
	Y Tilt	Deg.	-1.5	1.5	-2.2	2.2	-2.2	2.2	
	Total Tilt	Deg.	-	2.4	-	3.1	-	3.1	
	Rotation	Deg.	-1.0	1.0	-2.0	2.0	-2.0	2.0	
	DFOV	Deg.	72	76	72	76	72	76	
	55cm AFPos GCOL	-	65	125	63	127	50	150	GCOL
	55cm zPos	um	Data collection						
	Circle Center SFR	N/A	0.72	1	0.7	1	0.60	1	15% Degradation
	Circle 0.3F SFR sag/tan	N/A	0.56	1	0.54	1	0.46	1	
	Circle 0.6F SFR sag/tan	N/A	0.47	1	0.45	1	0.38	1	
	Circle 0.85F SFR tan	N/A	0.38	1	0.36	1	0.31	1	
	Circle SFR 0.3F Delta	N/A	-	0.2	-	0.23	-	0.27	Delta is based only defined ROI listed in chart below
	Circle SFR 0.6F Delta	N/A	-	0.20	-	0.23	-	0.33	
	Circle SFR 0.85F Delta	N/A	-	0.35	-	0.38	-	0.48	
Midgard	12cm zPos_FaceUp	um	-	-	Data collection				GCOL
	12cm zDeltaPos_FaceUp	um	-	-	Data collection				
PFL	PFL_error_um_12cm	um	-	-	-20	20	-27	27	
	PFL_error_um_55cm	um	-	-	-20	20	-27	27	
	GCOL_12cm_temp	C	-	-	18	55	18	55	
	GCOL_55cm_temp	C	-	-	18	55	18	55	
	12cm_Ze_top_linear_margin	um	-	-	5	-	5	-	
	55cm_Ze_linear_margin	um	-	-	Data collection				
	12cm_EFL	mm	-	-	Data collection				Temperate corrected
	55cm_EFL	mm	-	-	Data collection				
	12cm_Bz	DAC	-	-	Data collection				
	55cm_Bz	DAC	-	-	Data collection				
	12cm_PFL_expected	mm	-	-	Data collection				
	55cm_PFL_expected	mm	-	-	Data collection				
	12cm_di	mm	-	-	Data collection				GCOL
	55cm_di	mm	-	-	Data collection				
	12cm_do	mm	-	-	Data collection				
	55cm_do	mm	-	-	Data collection				
	12cm_do_inverse	1/mm	-	-	Data collection				
	55cm_do_inverse	1/mm	-	-	Data collection				
	12cm_do_inverse_err	mm	-	-	Data collection				GCOL vs expected
	55cm_do_inverse_err	mm	-	-	Data collection				
Black Level Offset	blackLevelOffset back	-	> -1	< 1	> -1	< 1	> -1	< 1	see section “Black Level Offset”

	F1	F2	F3	F4
Center	137	139	140	138
30F tan	79	87	187	195
30F sag	82	89	189	198
60F tan	46	66	210	230
60F sag	48	67	211	232
85F tan	1	14	263	276

0.3F Delta= Diff((30F1_tan+30F1_sag)/2, (30F2_tan+30F2_sag)/2, (30F3_tan+30F3_sag)/2, (30F4_tan+30F4_sag)/2)

0.6F Delta= Diff((60F1_tan+60F1_sag)/2, (60F2_tan+60F2_sag)/2, (60F3_tan+60F3_sag)/2, (60F4_tan+60F4_sag)/2)

0.85F Delta= Diff(85F1_tan, 85F2_tan, 85F3_tan, 85F4_tan)



Black Level Offset Calculation for IN Sensor

Excerpt from Ashirwad B on test sequence:

The suggested sequence of events to implement Black Level Offset Test

- turn on camera
- enable streaming
- do 2-byte read of 0x0400 for IN sensor

A sample code in h11isp is:

```
on
v
start 0 119 0
msecdelay 200
i2cread 4 0x10 0x0400 2 2
q
```

Once the 0x0400 2-byte value is available, do the following:

signBit = 0x0400[15]
msb = 0x0400[14:8]
lsb = 0x0400[7:0]

BlackLevelOffset = [-1*signBit*2^15 + hex2dec(msb)*2^8 + hex2dec(lsb)]/32

Any unit which violates **-1 < BlackLevelOffset < 1** range should be screened.

Sensor NVM Specification

Overview

This section outlines the Sensor NVM map, component parameters, checksums, and integrity check requirements.

Indiana-A NVM Map

C2.0_IN_NVM_Map

NVM Component Parameters:

NVM version:	Defined by Apple, when this NVM table changed, increase the version by 1. See VSR (Vendor Specific Requirements) for details.
Camera Project:	Unique project identifier. See VSR for details.
Integrator:	Integrator ID. Assigned by Apple. See VSR details.
Day:	Day of week as digital value of 1 through 7 with 1 being Monday, 7 being Sunday.
Work Week:	Digital value of 1 through 53.
Year:	Last digit of calendar year (9 for 2009, 0 for 2010, …)
Sequence Number:	Module Sequence Number. Each day the number will start at 0 and increment by 1 digital count for each module. At the exact time the day field changes the sequence number will reset to 0. So a combination of day, week, year, and Sequence Number results in a uniquely identifiable module. The Sequence Number is synchronized with the serial number in the barcode by the method of translation from 4 digit base 34 number to a 3 byte number based on the example in Table 18. Base 34 consists of the digits 0 through 9 and the letters "A" through "Z," excluding the letters "I" and "O" because of their similarity to the digits 1 and 0.
Actuator:	Actuator ID. Assigned by Apple. See VSR (vendor specific requirement) document for details.
Lens:	Lens ID. Assigned by Apple. See VSR (vendor specific requirement) document for details.
Driver:	AF driver ASIC. Assigned by Apple. See VSR (vendor specific requirement) document for details.
IRCF:	IR-cut filter configs. Assigned by Apple. See VSR (vendor specific requirement) document for details.
Substrate :	Substrate design ID. Assigned by Apple. See VSR (vendor specific requirement) document for details.
Sensor:	Sensor ID. Assigned by Apple. See VSR (vendor specific requirement) document for details.
Flex:	Flex design ID. Assigned by Apple. See VSR (vendor specific requirement) document for details.
Stiffener :	Stiffener design ID. Assigned by Apple. See VSR (vendor specific requirement) document for details.
Trim:	Alignment Trim ID. Assigned by Apple. See VSR (vendor specific requirement) document for details.
DoE Lookup:	Design of Experiment tracking. Assigned by Apple. See VSR (vendor specific requirement) document for details.
Process Control Plan Revision:	Control plan revision tracking. Any updates to process control plan made to a config to be updated. Control plan revision tracking required
Camera Build:	Module camera build stage. Used for tracking module maturity. Assigned by Apple. See VSR (vendor specific requirement) document for details.
Config Number:	Module build config number. Assigned by Apple. See build matrix for details
Test Station 1-5 ID:	Integrator test station ID tracking. The test station ID of all stations used to test a module shall be programmed. See VSR (vendor specific requirement) document for details.
Test Software Revision:	Test software revision tracking. Any updates to process control plan made to a config to be updated. Control plan revision tracking required
Integrator NVM Checksum:	Checksum to ensure Integrator NVM is intact. Checksum = (256 - (sum(0x00~0x17) & 0x0FF)) & 0x0FF Full byte.
AF Cal Checksum:	Checksum to ensure AF calibration NVM is intact. Checksum = (256 - (sum(0x20~0x2E) & 0x0FF)) & 0x0FF Full byte.
Color Cal R/G, B/G Light Source 1/2:	12-bit R/G, B/G color ratios for light source 1 and 2. (lower 11 bits of fraction). Refer to Appendix F for color calibration details.
Color Calibration Checksum:	Checksum to ensure color calibration is intact. Checksum = (256 - (sum(0x30~0x3E) & 0x0FF)) & 0x0FF Full byte.

VCM Barcode:	16-digit barcode shall be scanned and decoded into ASCII format according to the table 22.
Waiver Field:	Reserved for designating property not meeting full spec. Assigned by Apple. See VSR (vendor specific requirement) document for details.
X, Y center offset:	OCX and OCY programmed. Negative numbers are programmed as two's complement.
Color Shading Valid:	To indicate whether Color Shading Calibration is present and verified. See VSR for details.
Color Shading Checksum:	Checksum to ensure Color Shading NVM is intact. Checksum = (256 - (sum(0x63~0x2F6) & 0x0FF)) & 0x0FF. Full byte.

BCMS (Unique Serial Number) from NVM contents

INFO TYPE	Plant Code	Date Code of Manufacture at Supplier	Sequence Number	Config Code	Checksum
Format	PPP	YWWD	SSSS	EEEER	X
Number of Characters	3	4	4	5	1
Explanation	**PPP = Vendor and Plant/Factory Location:** The code indicating the Image Sensor Module vendor and where it is manufactured. This code will be assigned to the Image Module vendor by Apple.	**Y = Year:** 2 = 2012 3 = 2013 5 = 2015 etc… **WW = Week of Manufacture:** 01 to 53 weeks **D = Day** Days 1 to 7 with 1 = Monday	**SSSS = 4 character Sequence Number** (base-34) Each module must have a unique sequence number for each plant and day This allows for 34^{4} = 1,336,336 units per day per plant	**EEEE = Module Config Codes** The code indicating the Image Sensor Module Lens, and Sensor This code will be assigned to the Vendor by Apple. R is revision assigned by Apple in VSR.	**X = Checksum** See Appendix N for checksum calculation method
Example	XYZ605200Z3A2341V				
Example Meaning	XYZ	1052	00Z3	A2341	V
	Built at Vendor XYZ Inc. Singapore Plant	Manufactured on Tuesday of the 5th week of 2011	Sequence # 00Z3	Apple provided	SN checksum

BCMB Definition

BCMB is composed of the Bytes 0/0x00 to 0/0x17 inclusive of the NVM.

BCMB	Starting	Ending	Character Length
Byte Index	0	23	
Byte Address	0x00	0x17	48

NVM Checksums

The following calculations should be used to generate and verify NVM checksums. Checksums should be verified at the final test station on the production line as well as at OQC.

Integrator NVM Checksum [7:0]

checksum = (256 - (sum(0x00~0x17) & 0xFF)) & 0xFF	*generate checksum*
sum(0x00~0x17) != 0	*verify nvm exists*
(sum(0x00~0x17, 0x1F)) && 0xFF == 0	*verify checksum*

Color Cal Checksum [7:0]

checksum = (256 - (sum(0x30~0x37) & 0xFF)) & 0xFF	*generate checksum*
sum(0x30~0x3F) != 0	*verify nvm exists*
(sum(0x30~0x3F) & 0xFF == 0	*verify checksum*

Color Shading Checksum [7:0]

checksum = (256 - (sum(0x63~0x2F6) & 0x0FF)) & 0x0FF	*generate*
sum(0x63~0x2F7) != 0	*verify nvm exists*
(sum(0x63~0x2F7) & 0x0FF) & 0x0FF == 0	*verify checksum*

Process Checksum [7:0]

checksum = (256 - (sum(0x3FA~0x45D) & 0x0FF)) & 0x0FF	*generate checksum*
sum(0x3FA~0x45D) != 0	*verify nvm exists*
(sum(0x3FA~0x45D) & 0x0FF) & 0x0FF == 0	*verify checksum*

Override (AF/Color Cal) Checksum [7:0]

checksum = (256 - (sum(0x40~0x4E) & 0x0FF)) & 0x0FF	*generate*
sum(0x40~0x4F) != 0	*verify nvm exists*
(sum(0x40~0x4F) & 0x0FF) & 0x0FF == 0	*verify checksum*

NVM Override for AF Cal/Color Cal

In the event NVM data for AF calibration and/or color cal requires reprogramming, an additional bank of NVM registers is available. Approval to apply override on any finished goods requires approval by Apple and will be evaluated on a case-by-case basis.

Override is enabled if the *Override Color Cal/AF* status register is set to the following:

Override Color Cal/AF [1:0]	Value (bin)	Value (hex)
[0] Override Color Cal Fields, use 0x41-48 (logical OR mask)	0000 0001	0x01
[1] Override AF Cal parameters, use 0x49-4E (logical OR mask)	0000 0010	0x02

NVM Integrity Check

NVM Address	bits	Parameter	Min	Max	Notes
0x22	[7:0]	Camera Project	34	34	Indiana
0x23	[7:0]	Project Version	3	3	3: IN-C
0x24	[7:3]	Integrator	1	1	
	[2:0]	Plant Code	0	0	
0x25	[7:0]	Camera Build	30	30	
0x26	[7:0]	Config Number	1	255	
0x32	[7:5]	IRCF vendor	1	1	
	[4:2]	IRCF Revision	1	1	
	[1:0]	IRCF Variant	0	0	
0x33	[7:5]	Substrate vendor	1	1	
	[4:2]	Substrate Revision	1	1	
	[1:0]	Substrate Variant	0	0	
0x34	[7:5]	Sensor vendor	1	1	
	[4:2]	Sensor Revision	1	1	
	[1:0]	Sensor Variant	0	0	
0x35	[7:5]	Actuator vendor	2	2	
	[4:2]	Actuator Revision	1	1	
	[1:0]	Actuator Variant	0	3	
0x36	[7:5]	Lens vendor	1	1	
	[4:2]	Lens Revision	1	1	
	[1:0]	Lens Variant	0	0	
0x37	[7:5]	AF Driver vendor	2	2	
	[4:2]	AF Driver Revision	1	1	
	[1:0]	AF Driver Variant	0	0	
0x38	[7:0]	Sphere Sensor	0	0	Not used
0x39	[7:0]	APS Sensor	0	0	Not used
0x3A	[7:3]	Flex vendor	2	8	
	[2:0]	Flex Variant	3	7	
0x3B	[7:3]	Stiffener vendor	6	6	
	[2:0]	Stiffener Variant	1	1	
0x3C	[7:0]	Trim vendor	0	0	Not used
Sensor NVM					
0x17DB-0x17DC	[15:0]	NVM_Sensor_SlaveAdd	> 0	-	Data collect and record in test log
0x17E0 - 0x17E1	[15:0]	NVM_Sensor_Model	> 0	-	
0x17E2		NVM_Sensor_TOP	> 0	-	
0x17E8		NVM_Sensor_BOT	> 0	-	
0x17EF		NVM_Sensor_TestRev	> 0	-	

Appendix A - Base-34 Conversion

The integer ID number from the NVM is synchronized with the serial number in the barcode by the method of translation from a 3 byte integer to a 4 digit base 34 number based on the example in the table below. Base 34 consists of the digits 0 through 9 and the letters “A” through “Z,” excluding the letters ”I” and “O” because of their similarity to the digits 1 and 0.

Table: NVM values for SN ‘XSW341200Z3A2341’

Bank (dec)	Index	Byte (Hex)	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0x000	0x56							
0	1	0x001	0x18							
0	2	0x002	0x2F							
0	3	0x003	0x85							
0	4	0x004	0x28							
0	5	0x005	0xCC							
0	6	0x006	0x85							
0	7	0x007	0x1B							
0	8	0x008	0x3F							
0	9	0x009	0x8C							
0	10	0x00A	0x4B							

Table: Base-34 Conversion Example

Hex Value			Base-34 Value		
1015A0			**STVA**		
Digit Position	Digit	Value	Character Position	Character	Value
1	1	1048576	1	S	1021904
2	0	0	2	T	31212
3	1	4096	3	V	986
4	5	1280	4	A	10
5	A	160			
6	0	0			
Sum in Decimal	1054112		Sum in Decimal	1054112	

```
% Base-34 SN conversion, from a 11-byte NVM value string to a 17-byte SN.
% Also verify the checksum in parallel.
% Matlab function
function [SN, CS] = NVM2SN(NVM_value_hex)
%input:
% 'NVM_value' 11 bytes hex string, 3 sets of base-34 numbers for
% PPPYWW, DSSSS and EEEERX separatedly (X is the checksum digit).
% output: 'SN' 17 bytes SN (PPPYWWDSSSSEEEERX) string.
% CS: checksum. 'true' if checksum is correct, otherwise ‘false'.
% Initial Author: Shizhe Shen
% Questions to: ryan_j_dunn@apple.com
% last update: Sep. 29, 2014 to reflect updated contact information

NVM_length = length(NVM_value_hex);

% check the input string
if NVM_length ~= 22
  error('Please input an NVM Hex values of 22 digits (combining of 0 ~ 9, A ~ F)');
end

base34digits = '0123456789ABCDEFGHJKLMNPQRSTUVWXYZ';

% convert NVM value string to SN
dec_PPPYWW = hex2dec(NVM_value_hex(1:8));
dec_DSSSS = hex2dec(NVM_value_hex(9:14));
dec_EEEERX = hex2dec(NVM_value_hex(15:22));

SN = [];
for ii = 6:-1:1
   SN_temp = floor(dec_PPPYWW/34^(ii-1));
   dec_PPPYWW = dec_PPPYWW - SN_temp*34^(ii-1);
   SN = [SN,base34digits(SN_temp+1)];
end

for ii = 5:-1:1
   SN_temp = floor(dec_DSSSS/34^(ii-1));
   dec_DSSSS = dec_DSSSS - SN_temp*34^(ii-1);
   SN = [SN,base34digits(SN_temp+1)];
end

for ii = 6:-1:1
   SN_temp = floor(dec_EEEERX/34^(ii-1));
   dec_EEEERX = dec_EEEERX - SN_temp*34^(ii-1);
   SN = [SN,base34digits(SN_temp+1)];
end


total = 0;
% calculate checksum and verify
for ii = 1:16
   % convert to base-34
   digit = SN(17-ii);
   foundDigit = strfind(base34digits,digit)-1;

   % checksum calculation
   value = 34 - foundDigit;
   if mod(ii,2) == 0
      total = total + value;
   else
      total = total + 3*value;
   end
end
X = base34digits(mod(total,34)+1);

CS = (strcmp(X,upper(SN(17))));
```

Appendix B - Serial Number Checksum Code

The following code is written in C and consists of three (3) parts:

a) Main.c is the parent level file for checksum script

b) CheckDigitTest.h is a subroutine

c) CheckDigitTest.c is the logic in C

Main.c

```
#include <stdio.h>
#include "CheckDigitTest.h"

int main (int argc, const char * argv[]) {
    if (argc == 1) {
        // No parameters, nothing to test
        puts("No serial numbers to verify");
    } while (--argc > 0) {
        char serialNumber[20]; // storage space for serial number
        if (strlen(*++argv) > 18) {
            printf("'%s' is too long to test", *argv);
            continue;
        }
        int sourceValid = verifyCheckDigit(*argv); // Test just verify routine
        strcpy(serialNumber, *argv);
        int destGenerated = addCheckDigit(serialNumber); // Add check digit
        int destValid = verifyCheckDigit(serialNumber); // This test should always succeed
        printf("%s (%d): %s (%d,%d)\n", *argv, sourceValid, serialNumber, destGenerated, destValid);
    }
    return 0;
}
```

CheckDigitTest.h

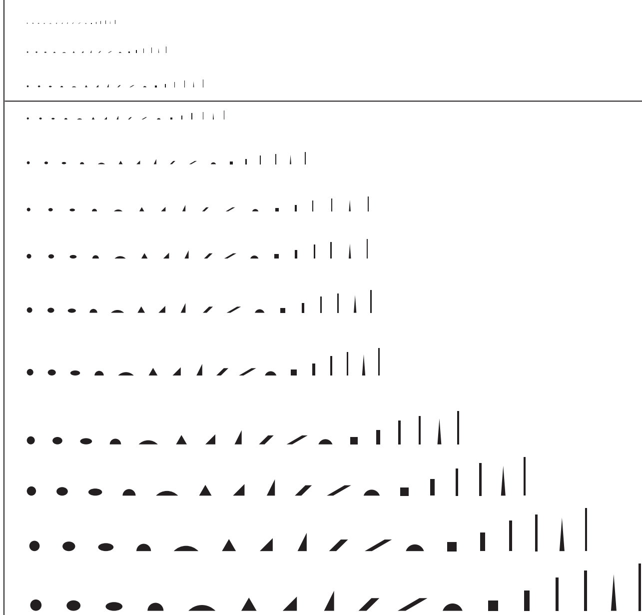
```
/*
 * CheckDigitTest.h * CheckDigitTest
 *
 * Copyright (c) 2005 Apple Computer Inc. All rights reserved.
 *
 */
#include <strings.h>
```

CheckDigitTest.c

```
/*
 * CheckDigitTest.c
 * CheckDigitTest
 *
 * Copyright (c) 2005 Apple Computer Inc. All rights reserved.
 *
 */
#include "CheckDigitTest.h"
/*
 * Compute check digit for Apple serial number. Check digit is appended to end of passed in string.
 * @param serialNumber null terminated serial number to add check digit to. Must contain enough room to
append another character.
 * @return true if serialNumber now has a valid check digit, false if check digit cannot be computed
 */
int addCheckDigit(char *serialNumber) {
    static char digits[] = "0123456789ABCDEFGHJKLMNPQRSTUVWXYZ";
    int length = strlen(serialNumber);
```

```
    int radix = 34; // always base 34
    int total = 0; // Start total at 0
    int index; // loop counter

    // Loop over characters from right to left with the rightmost character being odd
    for (index = 1; index <= length; ++index) {
        char digit = serialNumber[length - index];
        char *foundDigit = strchr(digits, digit);
        if (!foundDigit) { // Invalid digit, check digit can't be calculated
            return 0;
        }
        int value = foundDigit - digits;
        if ((index & 1) == 1) { // odd digit, add 3 times value
            total += 3*value;
        } else { // even digit, just add value
            total += value;
        }
    }

    // Compute and append check digit
    int checkValue = total % radix;
    char checkDigit = (checkValue > 0) ? digits[radix - checkValue] : '0';
    serialNumber[length] = checkDigit;
    serialNumber[length+1] = '\0';
    return 1;
}


/*
 * Verify check digit for Apple serial number.
 * @param serialNumber serial number to verify
 * @return true if serialNumber has a valid check digit, false otherwise
 */
int verifyCheckDigit(const char *serialNumber) {
    static char digits[] = "0123456789ABCDEFGHJKLMNPQRSTUVWXYZ";
    int length = strlen(serialNumber);
    int radix = 34; // always base 34
    int total = 0; // Start total at 0
    int index; // loop counter

    // Loop over characters from right to left with the rightmost character being even
    for (index = 0; index < length; ++index) {
        char digit = serialNumber[length - index - 1];
        char *foundDigit = strchr(digits, digit);
        if (!foundDigit) {
            // Invalid digit, check digit can't be calculated
            return 0;
        }
        int value = foundDigit - digits;
        if ((index & 1) == 1) {
        // odd digit, add 3 times value
        total += 3*value;
        } else { // even digit, just add value
            total += value;
        }
    }
    // verify that total is an even multiple of radix
    return (total % radix) == 0;
}
```

Appendix C - Cosmetic Defect Classification

SIZE (mm^2)	DEFECT SHAPE	A+	A	B	C
0.02		2	2	OK	OK
0.05		1			
0.08		NG	1		
0.1			NG	NG	
0.2					NG
0.3					
0.5					
0.7					
1.0					
1.5					
2.0					
2.5					
3.0					

1. Inspection Methods (condition)
 - 1.1 Light source: 500Lux - 700Lux across entire inspection area.
 - 1.2 Viewing distance: 30cm +/- 2cm
 - 1.3 Part rotation angle during inspection
   - 1.3.1 Vertical rotatation angle: +/- 75˚ from normal (top to bottom)
   - 1.3.2 Horizontal rotation angle: +/- 75˚ from normal (left to right)
 - 1.4 Viewing time: For front and back surfaces: 4 to 5 seconds (each side 2 seconds)

2. Cosmetic
 - 2.1 Defect types: scratch, molded internal contamination, black dot/line, air hole, pit, dent, crack, particle.
 - 2.2 Defect count: parts classified as Class A+ are allowed up to have 2 defects of size 0.02mm^2 max OR 1 defect of size 0.05mm^2 max, assuming defects spacing conforms with spacing prescribed in 2.3. the same 'OR' rule applies for Class A defects accordingly.
 - 2.3 Defect spacing
   - 2.3.1 10mm minimum spacing between one black spot and one spot of another color OR two defects of similar appearance equal to or less than 0.02mm^2 max for Class A+. Similar rule applies to Class A.
   - 2.3.2 30mm minimum spacing between two black spots or two hairline scratches, or two other defects of similar appearance.
   - 2.3.3 2.3.1 and 2.3.2 needs to meet Class A for each side (surface).
 - 2.4 Unless otherwise specified in cosmetic part drawing, use 25% contrast standard for inspection of visible hairline scratch (<0.01mm depth), knit lines, gate blush, and stress whitening. Do not use 25% contrast standard for inspection of LCD area.
 - 2.5 If surface contamination on parts cannot be cleaned, then part must be rejected. Clean with soft lint-free cloth and water only. 99% ethyl alcohol, isopropyl alcohol, and n-Heptane can be used on cosmetic surfaces if approved by Apple PD Engineering. Other cleaning agents must be approved by Apple PD Engineering.
 - 2.6 All cosmetic surfaces must be completely free of any fingerprints, smudge, dirt, adhesive residue, water spots, water marks, streaks or any remnant of incomplete cleaning. Cosmetic surface cleanliness must be maintained in packout.
 - 2.7 25% constrast standard should only be used per (2.4).

CONTRAST STANDARD

25%

SCALE

0 10 20 30 40 50 (mm)