First Light CASIMIR/GREAT observing modes and software implementation

J. Stutzki, H. Jakob, March, 28th, 2003, v1.5
(iterated with Sean Colgan, Stephan Stanko, ...)

1. Introduction

1.1. Philosophy

The two first generation heterodyne instruments on SOFIA, CASIMIR and GREAT, will use a common data acquisition and telescope control software for observing with SOFIA. This document describes the observing modes intended to be used from SOFIA-first light on. These are selected to be absolutely necessary for successful and efficient observing onboard the airborne observatory; at the same time, they are setup to be as simple as possible with regard to the observatory interface. Later on, more observing modes may be added, needing additional complexity at the telescope/observatory interface.

The observing procedures defined below are based on long term experience of observing at ground based radio telescopes and on the KAO observing experience. The observing procedures discussed are widely established at ground based mm- and submm-wave observatories such as CSO, JCMT, HHT and KOSMA. In fact, largely the same software will be used (together with a similar hardware setup) at some of those observatories quite some time before SOFIA flies. In particular, the identical telescope/observatory specific interface (telescope tracking and pointing commands , chopping secondary commands, and telescope/observatory status information) will be used. This ensures that the software is well understood and thoroughly debugged and has (at least some) heritage of successful real time use at an observatory by the time it gets used on SOFIA; potential problems are thus likely to be constrained to the immediate observatory/telescope interaction. This should give a good chance that they can be solved efficiently and without much of any loss of observing time on board SOFIA.

1.2. Software and hardware architecture

The observing software for both instruments will run on a (small) number of Linux-based PCs (and possibly also Linux-based VME-bus computers). It controls the instrument (front-end, backend, calibration system etc.) and the SOFIA telescope and secondary chopper, and does the data acquisition and archiving of the data in raw format. Quick-look and offline data analysis are a separate software package (although in the case of WASP, they are partly integrated with the data acquisition tasks) and making extensive use of the CLASS package. They access the raw and/or pre-reduced data provided by the observing software in the SDFITS format for raw data and CFITS for pre-reduced

The interface to the SOFIA telescope itself is through a TCP/IP- Ethernet connection via MCCS supported commands. These make the telescope track positions on the sky or scan along predefined tracks; and they control the chopper operation. They also feed back telescope and observatory status information to the observing system. Only the timing of

the chopper motion is directly controlled by a hardware TTL connection.

On top of this SOFIA/MCCS command layer is the CASIMIR/GREAT telescope task, connected to the observing software through a KOSMA_file_io-interface (see below) that is common to, and thus has heritage from, a number of other observatories.

1.3. Modularity

The observing software used by GREAT and CASIMIR is build in a modular way. There are many different ways to define modularity, so that a short description seems appropriate. Here 'modular' means that the various functions to be performed are distributed to relatively small, self-contained tasks, each of which acts under control of a set of input parameters and produces output in the form of further parameters used for input to other tasks or for status information. In the case of the immediate data acquisition tasks, the output contains also the data, generated in some raw format. These tasks internally communicate via the KOSMA_file_io system, i.e. through ASCII parameter files with time-stamps, that also allow for starting and stopping these tasks.

Each task can be run from the operating system command line level and its execution mode is entirely controlled by the KOSMA_file_io ASCII-files. More complex procedures are executed via shell scripts combining sequences of these low level tasks and creating/modifying the appropriate KOSMA_file_io-files. At yet a higher level, these tasks and shell scripts are ultimately controlled via a GUI-based observer interface.

This setup guarantees that every task can be debugged in a straightforward and transparent way, e.g. by changing its input files with any standard text editor. The few tasks that directly control pieces of hardware and thus need the physical presence of this hardware for functioning properly, can be replaced by 'dummy' tasks, that emulate the appropriate interaction with that particular hardware. In this way, the complete system, or any subset of the system, can be debugged in a standalone way without the need to have any of the instrument or observatory hardware attached.

2. Observing Modes

In the following, this document largely concentrates on describing the observing modes used by CASIMIR and GREAT at first light, and their detailed implementation. The presentation is oriented toward the software implementation of the defined observing modes and tries to be complete in the sense, that no additional information is necessary to fully specify the sequence of actions to be performed in order to execute an observing mode.

The observing modes are separated into a first, small set of 'fundamental modes', that are defined by a specific data acquisition and telescope/observatory operation mode. From these, a set of 'higher level' observing modes is constructed that allow to perform the full science functionality of the instruments/observatory.

The selection of the fundamental modes to be implemented is based on the need of doing

difference measurements on a time scale faster than the typical drift time scale of the instrument. The latter can extend out to some 100 seconds with substantial effort going into proper thermal control of all instrument components; but some 40 seconds or so are achieved more typically. Another important time scale entering into consideration is the typical integration time needed to reach a sufficient signal-to-noise ratio. Even for the brightest lines in the submm-/FIR-band relevant for SOFIA, present state of the art receiver sensitivities require some 10 seconds integration time to reach a S/N of a few to 20. This is of the same order as the drift time scales, implying that typically a few to many differencing cycles are necessary to reach sufficient S/N in a real observation.

Experience with heterodyne observing shows, that signal modulation on time scales of about 1-2 sec is always sufficiently fast; thus, modulating the signal with the secondary chopper (on and off the source) or with frequency-chopping at this rate is sufficient. This data acquisition mode is implemented as **signal-reference-switched sub-scans**.

As frequency chopping is limited to narrow frequency throws (defined by the smoothness of the instrument bandpass), its applicability is limited to sources with relatively narrow lines, typically of widths up to some 10 km/s. Sky-chopping with the secondary is limited to a maximum chop throw on the sky of typically a few arc-minutes (8' in the case of SOFIA) and is thus applicable only for sources with a small spatial extent. The only differencing technique that work for more extended sources and/or sources with wider spectral lines is a slow switch between on- and off-source position by moving the whole telescope. This motivates the implementation of the **total-power sub-scans**.

Total power observing typically requires relatively large overheads for the telescope movements (up to several seconds) and thus rapidly reduces observing efficiency for large on-off distances. If the extended source is to be mapped, the overhead can be reduced drastically by observing many on-source positions within one slow switch cycle together with a longer integration off-source. For optimum signal/noise, the off-source integration has to be *sqrt(n)* times longer than the individual on integration, where a whole cycle contains *n* on-positions. This concept leads to **on-the-fly (OTF) sub-scans** as another fundamental observing mode.

Although other fundamental observing modes may come into discussion, one should note that the above four cover all needs. For example, a sky-chopped OTF mode may at first sight look attractive to implement. With the telescope secondary chopping along the scan direction, one can derive the intensity distribution on the sky by integrating up the differentiated signal, as is regularly done today in broadband bolometer observations. However, comparison of the various time scales discussed above shows, that one can achieve the same and with similar efficiency by combining signal-reference-switched sub-scans at subsequent discrete positions. The issue here is the relatively long total integration time per position needed to achieve sufficient S/N (which is typically at least a few 10 seconds), compared to the overhead time involved in moving the telescope to subsequent near-by positions (which is typically about a second or so). Note that the data transfer time is similarly small, so that the data can be transferred while moving the telescope. In contrast, for bolometer observations of bright, extended sources, the per pixel integration time is on the order of only a second or so, so that the overhead for moving the telescope is a substantial fraction of the time, and on-the-fly observing is

more efficient.

Another observing mode widely in use at ground based facilities is total power raster mapping, where a small number *m* of subsequent discrete map positions is observed in one cycle with a *sqrt(m)*-longer off-source integration. We regard this as unnecessary to implement, as OTF observing is yet more efficient.

Based on these considerations, we do not, at least not for first light, aim at other fundamental observing modes, in particular as they would likely require a more complex synchronization interaction between the telescope, including the chopper, and the data acquisition.

2.1. fundamental modes

2.1.1. total power sub-scan (TP-mode)

In total power mode, the telescope is commanded to stare at a user defined position on the sky. After being commanded to go to that particular position, the telescope acknowledges arrival on that position and regularly feeds back status information confirming that it is still on track within a tolerance specified by the observing software. It keeps tracking that position until it is commanded to do otherwise.

The secondary chopper and all other signal modulation are switched off.

The instrument front-end has been tuned to a specified setting and regularly provides health (and other) status information

The backend is commanded to acquire data for a given length of time. It internally converts this length to the proper number of basic readout cycles (thus specifying the actual sub-scan duration, which may be shorter than the commanded one by a maximum of one readout cycle). It clears the add-up buffer and then integrates subsequent readouts to this buffer. As part of its output it may regularly output the number of readout cycles already integrated (this would serve as progress info). At the end of integration it raises an appropriate flag and returns the added-up buffer as integer-counts raw data for each spectroscopic channel.

The data acquisition can be aborted at any time (within a few readout cycles) by issuing an appropriate command parameter to the backend task.

Some additional information: due to the total stability time of typical heterodyne systems of not more than 100 seconds, TP-subscans will typically be of some 10 to 40 seconds duration on the sky. When used in calibration on the instrument internal hot and cold load (see bleow), they will typically be about 5 to 10 seconds, including possibly an equal length TP subscan on some blank sky position (for determining the atmospheric transmission). For spectrometer internal calibration (zero/comb, see below), TP-subscans of 2 seconds duration are typical.

NOTE: due the heterodyne instruments operating at relatively long wavelengths and with correspondingly large beam sizes, gyro tracking is fully sufficient. After an intial setup of the gyro inertial reference through one of the optical tracking cameras, SOFIA can be pointed blindly for up to several hours within an arcsec tolerance; resetting the gyro drifts is only needed on these long timescales by again using one of the tracking cameras. Therefore, both heterodyne instruments can operate with the solid aluminum tertiary, as long as they use the SOFIA images (WFI or FFI) or their internal CCD camera to set the gyros to the approriate intertial reference frame (CASIMIR has an internal CCD camera implemented, GREAT probably is not going to have one (tbd)).

2.1.2. signal-reference-switched sub-scan (S/R-mode)

In signal-reference mode, the telescope is commanded to track a user defined position on the sky (at least in one of the two sky-chopper positions, if sky-chopping). After being commanded to go to that particular position, the telescope acknowledges arrival on that position and regularly feeds back status information confirming that it is still on track within a tolerance specified by the observing software. It keeps tracking that position until it is commanded to do otherwise.

The secondary chopper or some other signal modulation are switched on, modulating the signal in phase with the S/R-TTL signal distributed throughout the system to the relevant hardware units. For chopping with the telescope secondary, chopper frequencies will typically be around a few Hz down to 0.5 Hz. Other signal modulation (e.g. frequency chop) will be of similar speed.

The instrument front-end has been tuned to a specified setting and regularly provides health (and other) status information

The backend is commanded to acquire data for a given, total length of time. It internally converts this length to the proper number of basic S/R-phases and the number of readouts per S- or R-readout cycles (thus specifying the actual sub-scan duration, which may be shorter than the commanded one). It clears the add-up signal and reference buffer and then integrates subsequent signal and reference readouts to these buffers. As part of its output it may regularly output the number of S/R-readout cycles already integrated (this would serve as progress info). At the end of integration it raises an appropriate flag and returns the added-up signal and reference buffers as integer-counts raw data.

The data acquisition can be aborted at any time (within a few readout cycles) by issuing an appropriate command parameter to the backend task.

2.1.3. on-the-fly sub-scan (OTF-mode)

In on-the-fly mode, the telescope is commanded to track along a great circle on the sky, starting at a user defined position and in a user defined direction and speed at a given start time in the (near) future, and for a given length. After being commanded to do so, the telescope acknowledges arrival on that position with the specified velocity (start of scan) and continues by regularly feeding back status information confirming that it is still on track within a tolerance specified by the observing software. After the specified length

of time, the telescope stops tracking at the given speed and keeps tracking the end position of the OTF sub-scan until commanded to do otherwise.

The telescope cannot, of course, reach the commanded position and track across it at the commanded velocity and at a specified time, starting from arbitrary positions on the sky. It would accordingly respond with “not on track”. In order to ensure efficient operation, the observing software has to command the telescope to track a ' pre-OTF' position, which is an appropriate distance in front of the OTF-sub-scan in order to ensure that the telescope can accelerate within its drive system physical torque limits to scan across the start position with the specified velocity at an approximately given time lap in the future. This distance of the ' pre-OTF' position is calculated by a observatory/telescope specific subroutine, that the observing software calls beforehand. Once tracking the ' pre-OTF' position, the observing software can command the telescope to start the OTF-sub-scan anytime, with the appropriate time lap between issuing the command and the start time specified. This procedure ensures that the telescope can successfully execute the OTF-sub-scan as long as it functions properly.

NOTE: interface to/calling syntax for this ' pre-OTF' subroutine needs to be specified!

The secondary chopper and all other signal modulation are switched off.

The instrument front-end has been tuned to a specified setting and regularly provides health (and other) status information

The backend is commanded to acquire data in ' continuous' readout and transfer mode, i.e. starting at the given start time in the (near) future, and integrating a specified (small) number of readout cycles into a series of subsequent data buffers, each buffer representing the accumulated signal for a position interval along the OTF-sub-scan. The buffers are cleared at the beginning of the sub-scan. While acquiring data, the backend software has to transfer the previously accumulated buffer to the data archive and data reduction pipeline as integer count raw data. As part of its output it may regularly output the number of readout buffers already finished (this would serve as progress info). At the end of integration it raises an appropriate flag to signal the end of the data acquisition.

Shortly past the specified start time, the observing task has to verify that both the telescope track and the data acquisition were started successfully; this is done by polling the KOSMA_file_io-feedback files from these subsystem.

The data acquisition can be aborted at any time (within a few readout cycles) by issuing an appropriate command parameter to the backend task.

Some additional information: due to the total stability time of typical heterodyne systems of not more than 100 seconds, OTF-subscans will typically be of total duration of order 40 to 60 seconds. In order to get acceptable low level of distortion of the resulting maps from the beam smearing due to the continuos motion, on the order of 3 to 4 readout cycles per beam diameter are necessary. In order to keep the data volume and data handling times in reasonable limits, the individual readout cycles are typically selected to be of 1 to 4 second duration, resulting in slew speeds of *(beam diameter)/(readous per*

beam diameter)/(length of readout cycle), which amounts to typical slewing speeds of maximum 20 arcsec/second to minimum 2 arcsec/second for observations of astronomical sources. On SOFIA, such OTF scans would thus be within the fine drive range. When OTF mode is used for a skydip (which on SOFIA implies using the coarse drive with lower positional tolerance, see below), one would typically use readout cycle times of some few seconds, each covering about a few degree in elevation range, and thus would need slew speeds in elevation of about 1 degree/second.

2.2. telescope/chopper-control interface

The fundamental observing modes detailed above largely define the parameters that are needed to control the telescope functions and that have to be communicated to the telescope and chopper control task via the specific KOSMA_file_io-ASCII parameter files. As the actual telescope position is usually derived from a number of other parameters like the source positions, map offsets, off-source offsets etc., there is still, however, a variety of choices with regard to which parameters one wants to explicitly hand over to the telescope control task. The selection presented below is guided by allowing the observer to easily trace back what position within a given observing project the telescope actually tracks. Thus, one should, for example, explicitly specify map offsets so that the observer immediately sees in a status display of the observing software/telescope task interface that the telescope presently tracks a given offset position relative to the source center position.

The feedback from the telescope/chopper, needed to acknowledge commands to those systems and to monitor the proper performance of the telescope and chopper, again goes via KOSMA_file_io-files. This file-interface is to be polled by the observers task after a command to e.g. the telescope was issued, until the execution of the command is acknowledged by the corresponding status flag. It may also be polled at any time after and should reflect changes of the status (e.g. "lost track"). The telescope feedback-file also contains a small set of parameters derived by the telescope/observatory, that are essential for the data acquisition to control and/or command other subsystem (such as the LO-settings, beam rotator angle) and that need information that only the telescope/observatory knows about; and it contains a few parameters for status information of the telescope subsystem (e.g. track errors).

NOTES:

a) the interface defined here is the ***control***-interface between the observing/instrument software and the telescope/observatory. It should thus be judged according to whether it

i) provides all the parameters necessary to control the telescope in the functionality that the instruments/observer need.

ii) gives all the necessary feedback from the telescop/observatory to confirm its proper performance.

It is **not** the ***housekeeping*** interface to the observatory. The latter will handle all the observatory/telescope status parameters that are required to provide full header information for the proper calibration of the data, but also additional information for proper documentation of the telescope/observatory status in the data archive.

b) the present version addresses only the fundamental telescope, i.e. pointing/tracking and chopper functionality. There will be more things in the observatory that need to be controlled by the instrument/observer: gate valve operation, telescope caging, focussing, nulling the gyros on a suitable optical reference position, ... It is not clear at present, what kind of interface MCCS will provide to the observer to control these functions. One option is to also include them in a separate KOSMA_file_io-interface. This would guarantee that we have a unique interface architecture between the observing task and the telescope.

		parameter	*encoding*	*definition*	*comments*
from observing task to telescope control task (file obs2tel):					
info					
		obs_source_name	%64s	source name	for display and archiving @ token to mark special objects like planets (Note 1)
		obs_scan_num	%12d	scan number	for display and archiving
		obs_sub_scan_num	%12d	subscan number ' '	for display and archiving
		obs_tel_info_update_time	%12.8g	cycle time for updating feedback-file [seconds	= 0: no continuous update
		obs_cookie	%12d	unique internal identifier for observation	is fed-back with info, status and error parameters to identify the observation they refer to
telescope positioning					
	source position	obs_coord_sys_on	%32s	coordinate system of source position	list of allowed coordinate systems: Note 2
		obs_lam_on	%12.8g [degree]	first coordinate	
		obs_bet_on	%12.8g [degree]	second coordinate	
	map position (static):	obs_coord_sys_del	%32s	coordinate system of map offsets	list of allowed coordinate systems: Note 2
		obs_true_angle_del	%1s [Y/N]	apply cos(beta)-correction for map offsets	(del_lam_MCS = obs_lam_del/cos(obs_bet_del) CHECK!)
		obs_lam_del	%12.8g [arcsecs]	map offset, first coord.	
		obs_bet_del	%12.8g [arcsecs]	map offset, second coord.	

		parameter	*encoding*	*definition*	*comments*
	map position (velocities)	obs_lam_vel	%12.8g [arcsec/sec]	scan speed, first coord.	note: include cos-correction if requested!
		obs_bet_vel	%12.8g [arcsec/sec]	scan speed, second coord.	
		obs_otf_mode	%1s [Y/N]	flag for OTF mode	
	off-position	obs_coord_sys_off	%32s	coordinate system for OFF position	list of allowed coordinate systems: Note 2
		obs_true_angle_off	%1s [Y/N]	apply cos(beta)-correction for OFF position	
		obs_lam_off	%12.8g [arcsec]	relative OFF position, first coord.	
		obs_bet_off	%12.8g [arcsec]	relative OFF position, second coord.	
	chopper position	obs_coord_sys_chop	%32s	see obs2chop	in obs2tel, these first three parameters do not control the secondary chopper, but specify proper telescope tracking
		obs_amp_chop	%12.8g [arcsec]	see obs2chop	
		obs_angle_chop	%12.8g [degree]	see obs2chop	
		obs_chop_beam	%d	specifies chopper beam to track	=+1: +-beam =-1: - beam =0: chopper is off (amp=0) (=0 also used for old KOSMA/AOSOBS mode backward compatibility, where off_lam, off_bet specify beam position)

		parameter	*encoding*	*definition*	*comments*
	focal plane nominal position (boresight) and orientation	obs_coord_sys_focal_plane	%32s	coordinate system for focal plane (TELESCOPE/ HORIZON)	HORIZON is the traditional radio telescope Nasmyth mount, requiring the telescope tracking to account for the elevation dependent Nasmyth-corrections; TELESCOPE is a Cassegrain mount or the SOFIA mount, where the instrument is mounted fixed relative to the telescope optics **AI:** define x,y directions consistent with SOFIA conventions
		obs_x_focal plane	%12.8g [arcsec]	referecence position in focal plane, first coord	instrument internal coordinates (move with beam rotator)
		obs_y_focal_plane	%12.8g [arcsec]	reference position in focal plane, second coord.	
		obs_los_angle_focal_plane	%12.8 [degree]	"line of sight" (los) rotation angle for focal plane	(+- 3 degree on SOFIA, 0 everywhere else **AI:** define 0 degree and orientation
		obs_los_mode	%5s	modus for los-tracking SET/TRACK	SET: set telescope to commanded los angle and stay there; TRACK: set telescope to commanded los angle and track this sky orientation
control parameters					
		obs_track_duration	%12g [seconds]	duration of scan	>0 in OTF mode, =0 permanent tracking
		obs_start_time	%20.5f [seconds]	start of scan [seconds from 1970.00 UT]	gives time in (near) future at which data acquistition starts, i.e. telescope should be on track
		obs_tolerance	%12.8g [arcsec]	maximum allowed tracking error	

		parameter	*encoding*	*definition*	*comments*
from observing task to secondary chopper control task (file obs2chop):					
	info	obs_cookie	%12d	internal identifier	identical to cookie in obs2tel
		obs_chop_info_update_time	%12.8g	cycle time for updating feedback-file [seconds	= 0: no continuous update
	chopper positioning	obs_coord_sys_chop	%32s	coordinate system for chopper positioning	see Note 1)
		obs_amp_chop	%12.8g [arcsec]	secondary chopper amplitude	note: amplitude is half the throw (CHECK NOTATION)
		obs_angle_chop	%12.8g [degree]	secondary chopper angle	ccw from positive direction of first coordinate
		obs_frequ_chop	%12.8g [Hz]	chopper frequency	although the actual chopper signal is fed to the chopper from the instrument as a TTL signal; the chopper needs this info in order to select the appropriate control loop/filter parameters for the chopper drive.
	status	obs_chop_active	%1s [Y/N]	switch chopper on or off	
from telescope back to observing task (file tel2obs):					
(updated when status and info parameters change and at the time interval specified by obs_info_update_time)					
	status:	tel_telescope	%32s	Telescope Identifier	(for display)
		tel_on_track	%1s [Y/N]	Y if tracking on commanded position/track within tolerance	timing: on new-position-command, telescope has to set "off track" for old track a.s.a.p; obs-task has to wait for tbd time interval before polling status
		tel_lost_track	%1s [Y/N]	Y if tracking got beyond tolerance since start of track	

		parameter	*encoding*	*definition*	*comments*
	error:	tel_pos_in_range	%1s [Y/N]	Y if commanded position is within telescope range	
		tel_error	%12d	flags telescope command syntax/consistency and functional errors	=0: ok other: error code (definitions to be done)
	info:	tel_return_cookie	%12d	return cookie to identify proper observation	
		tel_latitude	%12.8g [degree]	latitude of observatory	needed for Doppler correction
		tel_longitude	%12.8g [degree]	longitude of observatory [+west]	needed for Doppler correction
		tel_altitude	%12.8g [meter]	Height above sea level	needed for Doppler correction
		tel_angle_focal _plane	%12.8g [degree]	angle between on_source coordinate system and focal plane	ccw angle from first coordinate of focal plane system to first coordinate of on_source-system
		tel_los_act	%12.8g [degree]	actual los-angle	needed to check when to reset tracking in TRACKING los-mode.
		tel_time_act	%20.5f [seconds]	Time when status data were valid	
		tel_time_del	%12.8g [seconds]	Time since start of track	(may be negative, if start in the future)
		tel_azm_cmd	%12.8g [degree]	commanded azimuth	for tracking display
		tel_elv_cmd	%12.8g [degree]	commanded elevation	for tracking display and for atmospheric calibration
		tel_azm_act	%12.8g [degree]	actual azimuth	for tracking display

		parameter	*encoding*	*definition*	*comments*
		tel_elv_act	%12.8g [degree]	actual elevation	for tracking display
from chopper control back to observers task (chop2tel):					
(updated updated when status parameters change and at the time interval specified by obs_chop_info_update_time)					
	status	chop_return_cookie	%12d	return cookie to identify proper observation	
		chop_error	%12d	flags chopper command syntax/consistency and functionality error	=0: ok other: error code (definitions to be done)
		chop_ok	%1s [Y/N]	chopper is presently performing ok	
		chop_failed	%1s [Y/N]	chopper failed sometime since last commanded	

Note 1: @ as a prefix to the source name specifies a moving target, whose position has to be derived e.g. from ephemeris. "@" overrides the on source coordinates. The definition of what are allowed object names will not be given here and depend on the implementation of the telescope control software and hardware allowances.

Note 2: The following coordinate systems will be allowed: B1950, J2000, GALACTIC, HORIZON, (others?). This holds for on_source, map_offsets, offset_coordinates, chopper coordinates (within the allowances of the hardware).

2.3. higher level observing modes

2.3.1. intensity calibration

2.3.2. without sky transmission

2.3.3. with sky transmission

2.3.4. frequency calibration

2.3.5. sky-dip

2.3.6. total power scan

2.3.7. double beam-switch scan

2.3.8. OTF-scan

2.3.9. extended mapping